UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA

AudioMood: Classificação de emoções em bandas sonoras de filmes usando Redes Neuronais

Francisco de Andrade Bravo Mendonça

MESTRADO EM INFORMÁTICA

Dissertação orientada por:
Prof. Doutor Thibault Langlois

2021

Agradecimentos

Esta dissertação demorou bem mais do que devia, e não seria possível sem a paciência e disponibilidade do Professor Thibault Langlois, que foi uma grande ajuda desde o ínicio.

Sempre foi dito que é preciso uma aldeia para educar uma criança, e esta criança não é diferente. Desde dos meus pais, que sempre me encorajaram a prosseguir os estudos; á minha namorada Hanna que sempre me ajudou no que foi preciso ; aos meus colegas que sempre me desafiaram e trouxeram o melhor de mim.

Existem muitas mais palavras por dizer, mas sendo eu um péssimo escritor há apenas uma que vale a pena: Obrigado, a todos.

Para a minha mãe

Resumo

O recurso à Inteligência Artificial para a ajuda ou execução de uma tarefa é cada vez mais frequente na nossa vida. Desde assistentes pessoais e médicos ou até carros autónomos, o uso é vasto e é adoptado nas mais diversas áreas. Com o aumentar de complexidade das AI, estas requerem a criação de novos métodos para melhorar o treino de tarefas complexas. Nesse sentido, esta dissertação tenta ajudar o estudo dos métodos de treino de Redes Neuronais, utilizando áudio de modo a que a rede consiga identificar os sons presentes num filme. Para concretizar esse objectivo, o primeiro passo foi a análise de diversos *datasets*, de forma a seleccionar um que seja adaptado à metodologia utilizada. O *dataset* escolhido foi o AudioSet da Google, pois tem mais de dois milhões de vídeos anotados, algo que favorece este estudo. De seguida, foram desenvolvidas ferramentas para a criação de conjuntos mais pequenos de dados com base no AudioSet. Estas ferramentas trataram do *download* dos vídeos, a sua conversão em áudio, a manipulação e tratamento dos últimos, e a construção de novos *datasets*. No processo anteriormente descrito, foram aplicados os métodos de aumentação de dados, sendo estes a rotação de dados e o controlo de volume. Após a criação do *dataset* procedeu-se o treino. Para cada treino foi utilizado a mesma arquitectura do modelo, com pequenas diferenças no método de treino. É possível afirmar que para a tarefa escolhida, o aumento de dados no dataset e o uso de rotação de dados melhorou os resultados, enquanto a manipulação de volumes não ofereceu alterações suficientes aos dados para permitir que o modelo melhorasse.

Palavras-chave: Inteligência Artificial, Redes Neuronais, Augmentação de Dados, AudioSet, Datasets Abertos

Abstract

Nowadays the use of Artificial Intelligence to help or execute a task is ever more frequent. From personal assistants, to video games, to autonomous cars, the ability to use AI is vast, and getting adopted in new areas. As the complexity of AI increases, the necessity of developing new methods to help in the training of AI is critical. In that sense, this dissertation tries to help in the study training methods for Neural Networks, using audio sources, so that it is able to identify the different sounds present in a movie. To meet this purpose, the first step was the analysis of different datasets, to find one that is adaptable to the methodology used. The chosen dataset was AudioSet by Google, which has more than 2 million annotated videos. Later, tools were developed to create smaller datasets from AudioSet. These tools took care of video download, their conversion to audio, the manipulation and treatment of these audios, and the construction of new datasets. In this process, data rotation and volume control, two methods of data augmentation, were applied with the intention of creating new data. With the above mentioned new dataset, models were trained. The same model architecture was used for all the training processes, but with small differences in the training method. For the chosen task, it can be said that the increase of data in the dataset and the use of data rotation improved the test results, while volume control didn't offer sufficient alterations to the data, and so the test results didn't improve.

Keywords: Artificial Intelligence, Neural Networks, Data Augmentation, AudioSet, Open Datasets

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

A Inteligência Artificial (AI) é um tópico que há muito tempo tem interessado o ser humano. O primeiro interesse começou como um exercício de imaginação por parte de escritores, cineastas e outros. Em 1926 saí o filme *Metropolis*, considerado um dos primeiros filmes de ficção científica, onde um robô semi-humanoíde altera o seu comportamento levando a uma revolta dos trabalhadores. Foi uma primeira tentativa de representação de uma AI e a partir daí, por norma, na *pop culture* as AI são representadas de uma forma maléfica, sendo o seu ponto culminante a tentativa de destruir a raça humana, considerando-a desnecessária e contrária dos seus objectivos. Outras vezes, AI é retratado como um *side-kick* ou um meio que ajuda o humano a chegar ao seu objectivo. Muitos livros e filmes continuaram a explorar a temática de AI e da evolução tecnológica, o que proporcionou e continua a proporcionar os cientistas a deixar de olhar para o tópico como ficção científica e começar realmente a criar tecnologias mais inteligentes, evoluídas e sofisticadas, estudando mais o cérebro humano e tentar recriar a sua função por meios tecnológicos.

Um dos primeiros passos para tornar IA realidade ocorreu em 1943 com Warren McCulloch e Walter Pitts que conseguiram descrever pela primeira vez o que é uma rede neuronal biológica. A sua descoberta é publicada num artigo sobre como os neurónios funcionam, e como é possível a sua recriação simples utilizando circuitos eléctricos [15]. Este artigo inovador acelerou o desenvolvimento de duas áreas de pesquisa: processos biológicos no cérebro e a aplicação de redes neuronais em AI.

Ao longo dos próximos 16 anos, os cientistas continuaram a explorar esses tópicos, até que em 1959 David Hubel e Torsten Wiesel, demonstram o próximo grande avanço no mundo das redes neuronais. Eles conseguiram descrever a constituição do córtex visual, que é dividido em dois tipos de células: *células simples* e *células complexas* [8]. Esta descoberta ajudou em 1975, Kunihiko Fukushima descrever a primeira rede neuronal de multicamada, chamado Neocognitron [4]. O objectivo principal desta pesquisa era criar um sistema computacional capaz de resolver problemas como um cérebro humano. No entanto, com o passar do tempo, a dificuldade dessa tarefa tornou-se evidente, a tecnologia

da altura não estava evoluída para a ambição humana. Deste modo, o foco foi alterado para a criação de redes neuronais que tenham tarefas específicas, desviando-se de uma abordagem estritamente biológica.

Desde então, as redes neuronais têm oferecido suporte às diversas tarefas, desde as mais simples, como por exemplo, vídeo-jogos e alguns jogos de tabuleiro, como as mais complexas - reconhecimento da fala, diagnósticos médicos, e como temática desta dissertação - a classificação de emoções em bandas sonoras de filmes.

1.1 Motivação

As redes neuronais são idealmente desenvolvidas para nos ajudar a resolver problemas complexos em diversas situações da vida real. Estas podem aprender e modelar relações entre entradas e saídas de dados que são não-lineares e complexas; realizar generalizações e inferências; revelar relacionamentos, padrões e predições ocultas e modelar dados altamente voláteis (como dados de séries temporais financeiras) e variâncias necessárias para prever eventos raros (como detecção de fraudes). Como resultado, as redes neuronais podem melhorar processos de decisão em diversas áreas, como, por exemplo, área financeira, medicina, energia, entre outros.

Um dos problemas actuais das redes neuronais é a sua necessidade de grandes conjuntos de dados, para ter um bom desempenho. Assim, uma das soluções possíveis é o aumento de dados (*augmentation*), cujo objectivo é criar novos dados, usando dados existentes. Logo, em teoria, será possível usar um conjunto de dados relativamente pequeno que, ao fazer *augmentation*, permite obter resultados optimizados do que aqueles obtidos pelo *dataset* original. Estas técnicas dependem do tipo de dados, e da correlação que os dados tem entre si. Por exemplo, se os dados utilizados são compostos por fotos, uma solução seria a rotação das fotos, ou a adição de ruído. Assim, o modelo seria mais capaz de generalizar as previsões.

A motivação desta dissertação é a construção, avaliação e estudo de técnicas e ferramentas que possam no futuro ajudar a criar ferramentas que sejam usadas para a detecção de emoções em bandas sonoras de filmes.

1.2 Objectivos

O objectivo principal deste trabalho, é o estudo da evolução de uma Rede Neuronal utilizando *datasets* abertos. Estes, estão sujeitos a mudanças ao longo do tempo, reflectindo assim o mundo físico.

Para alcançar esse objectivo, foram desenvolvidas algumas tarefas intermédias, de modo a facilitar o processo de estudo. Estas, são a análise e manipulação de *datasets*.

Primeiramente, analisam-se diferentes *datasets* de modo a encontrar um, cujos resultados possam ser validados. Após isso, criam-se *datasets*, com base num *dataset* ”mãe”. Isso significa criar conjuntos mais pequenos, que contêm parte da informação de um *dataset* maior, e, progressivamente, continuar a adicionar mais dados desse conjunto ”mãe”, de modo iterativo, consoante as necessidades do modelo.

No final, estudam-se métodos de aumentação de dados, para poder aumentar o número de dados em cada um dos *dataset*. Por fim, averigua-se se estes métodos são de facto eficazes neste tipo de *datasets*. A Rede terá de conseguir, com uma precisão satisfatória, prever quais são as ”emoções”contidas em cada troço de um filme.

1.3 Contribuições

Esta dissertação contribui com diferentes experiências sobre técnicas de construção de *datasets*, especificamente, *datasets* constituídos de áudios. Dentro destas técnicas de construção, foram experimentadas duas técnicas de aumentação de dados: rotação de dados e controlo de volume. Foram desenvolvidas ferramentas, para aplicar as técnicas de aumentação de dados em áudios. Por fim, esta dissertação contribui com uma avaliação dos *datasets* criados, utilizando dados ”reais”, ou seja, que dados não oriundos do mesmo *dataset*.

1.4 Estrutura do documento

Este documento está organizado da seguinte forma:

- Capítulo 1 - Introdução - neste capítulo é introduzido o tema, a motivação e os objectivos do projecto.
- Capítulo 2 – Trabalho Relacionado - este capítulo foca-se na descrição de estado de arte e as metodologias utilizada para cumprir os objectivos definidos.
- Capítulo 3 – Análise de *Datasets* - neste capítulo será descrito os *datasets* estudados, a sua análise, e as razões pela sua utilização ou nao.
- Capítulo 4 - Implementação - neste capítulo será descrito a arquitectura do modelo utilizado, a criação dos *datasets* de treino e de teste, bem como as técnicas de aumentação de dados.
- Capítulo 5 - Resultado e Discussão - neste capítulo será descrito os resultados obtidos, e será discutido as conclusões obtidas entre cada experiência.
- Capítulo 6 - Conclusão

Capítulo 2

Trabalho relacionado

2.1 Emoções

Passamos toda a nossa vida a interagir num ambiente social vasto e complexo. Todas as nossas acções giram em torno de interacções com outras pessoas. As emoções servem de pistas para estas interacções. As emoções expressas por um agente comunicam informações sociais a um observador. Sejam expressões faciais subtis de tristeza ou gestos corporais zangados para assinalar o domínio, somos especialistas em reconhecer a expressão das emoções através de uma vasta gama de situações. De uma forma mais simples e resumida, uma emoção é uma reacção a um estímulo ambiental ou cognitivo, que produz reacções químicas no nosso corpo [20].

As emoções diferem umas das outras ao longo de várias dimensões. Por exemplo, algumas emoções são ocorrências (e.g. pânico), e outras são disposições (e.g. hostilidade); algumas são de curta duração (e.g. raiva) e outras de longa duração (e.g. dor); algumas envolvem processamento cognitivo primitivo (e.g. medo de um objecto que se aproxima subitamente), e outras envolvem processamento cognitivo sofisticado (e.g. medo de perder um jogo de xadrez); alguns são conscientes (e.g. repugnância por um insecto na boca) e outros são inconscientes (e.g. medo inconsciente de falhar na vida); alguns têm expressões faciais prototípicas (e.g. surpresa) e outros carecem delas (e.g. arrependimento). Algumas envolvem fortes motivações para agir (e.g. raiva) e outras não (e.g. tristeza). Algumas estão presentes em todas as espécies (e.g. medo) e outras são exclusivamente humanas [28].

Esta variedade multidimensional levou alguns cientistas a concluir que, as categorias de emoções comuns, não designam tipos naturais, e que, são diferentes para cada tipo de pessoa mesmo tratando-se de emoções facilmente identificáveis, como raiva, aversão ou medo [31], [9], [27], [2]. Outros argumentaram que existe, no entanto, homogeneidade suficiente entre as emoções comuns [32]. Estas dimensões podem ser categorizadas em: Valência e Excitação (mais conhecido pelo nome inglês - *Arousal*) [11].

2.1.1 Valência e Arousal

Tal como referido em cima, para simplificar as emoções, podemos descrevê-las utilizando dois conceitos: Valência e *Arousal*. A Valência indica o grau de prazer sentido na emoção, ou seja, de um lado do espectro temos emoções felizes, e do outro, temos infelizes. O *Arousal* indica a "*força*" da emoção, isto é, indica o quão forte uma pessoa sente uma emoção. Ao juntar estas duas dimensões é possível descrever praticamente todas as emoções. James Russel afirma que estas dimensões são lineares e independentes, ou seja, um sentimento agradável, não indica nada sobre o quão calma ou activa uma pessoa está [25].

2.2 *Machine Learning*

Machine Learning (ML) é uma área de estudo dentro da computação, cujo objectivo primário é o desenvolvimento de algoritmos que desempenhem uma tarefa, sem que lhes seja dado instruções explícitas. Para que o algoritmo consiga desempenhar a tarefa, este infere padrões dos dados que lhe são apresentados. Um dos requisitos deste tipo de algoritmos é conseguir generalizar, bem o suficiente, o que aprenderam. Isso permite que, ao apresentar novos dados, diferentes daqueles que já conhecem, os algoritmos continuem a reconhecer os padrões aprendidos [18].

Os algoritmos provenientes de *machine learning* podem ser categorizados em dois tipos: aprendizagem supervisionada (AAS) e não supervisionada (AANS). Nos AAS é criada uma função que irá conseguir mapear os dados de saída com os dados de entrada, ao ser transmitido ao algoritmo um par de dados de entrada e a saída esperada. Portanto, o objectivo final do algoritmo é conseguir prever o resultado de cada instância [26]. Por outro lado, os algoritmos de aprendizagem não supervisionada procuram padrões que, não tinham sido descobertos num conjunto de dados. A particularidade é que, esta descoberta é feita quase sem nenhuma supervisão humana e, muitas vezes, os dados de entrada não estão classificados. Para criar os padrões necessários para a classificação de dados, estes algoritmos utilizam modelos de probabilidade, para organizar os dados em diferentes tipos de conjuntos. Um exemplo disto é o algoritmo *K-Means*, que tem como objectivo criar conjuntos de dados que sejam semelhantes, dividindo os dados disponíveis em K grupos. Para isto, introduz-se cada dado no grupo, em que a média do grupo fica mais perto do dado. Para escolher a que grupo inserir o dado, utiliza-se uma métrica que calcule a distância entre o dado e a centro do grupo. Este método não é supervisionado, visto que não existe intervenção humana [14].

2.3 Redes Neuronais

Redes neuronais são sistemas artificiais que tentam simular os neurónios do cérebro humano. Estas usam algoritmos, para reconhecer padrões escondidos e correlações entre muitos dados. As redes neuronais têm a capacidade de agrupar a informação recebida e classificá-la, e – com o tempo – aprender e melhorar continuamente. [3]

2.3.1 *Perceptrons*

O *Perceptron* foi o primeiro tipo de rede neuronal. A ideia principal deste, era imitar o comportamento de um neurónio humano. Para fazer isso, o *Perceptron*, tendo como *input* um *set* de binários, multiplica-os por um peso, que tem como valor um número contínuo. Após isso, o valores são postos contra um *threshold*, ou seja, se a soma de todos os *inputs* a que foram aplicados os pesos, for maior que o *threshold*, a rede retorna 1, se for menor retorna 0 [21].

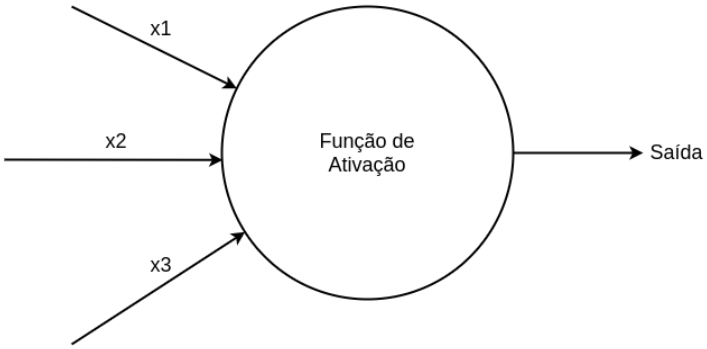

Figura 2.1: Exemplo visual de um *Perceptron*

O *Perceptron* tem como base um algoritmo supervisionado que classifica binariamente. Isso significa que, ao receber como *input* um vector de números, infere se esse *input* pertence a uma classe específica ou não. Este tipo de algoritmos são lineares, ou seja, fazem previsões com base numa função de previsão linear. Este tipo de previsão combina um conjunto de pesos, com o vector de características, que neste caso é um vector de números [22].

Sendo que se trata de um algoritmo supervisionado, para treinar um *Perceptron* é necessário passar pares *input-output* de exemplo, onde para cada dado de entrada existe a classificação correspondente. Assim, após a classificação dos dados, o algoritmo pode comparar a sua resposta com o resultado desejado, ajustando os seus parâmetros.

Contudo, existe uma limitação deste tipo de rede neuronal - os *Perceptrons* apenas permitem o uso de funções lineares. Isso é um problema quando existe uma situação que não pode ser restringida a cálculos lineares, como por exemplo uma operação lógica XOR. Nesta situação, como evidenciado pela Figura 2.2 , o *Perceptron* não consegue formar a área em que todos os casos positivos são bem classificados [16].

Figura 2.2: Problema XOR - *Perceptron*

2.3.2 *Multilayer Perceptrons*

Para resolver a limitação referida anteriormente, foi criado um tipo de rede neuronal que utiliza diversos *Perceptrons*, designado por *Multiplayer Perceptrons* (MLP). Estes, estão dispostos em várias camadas, ligadas entre si, formando um grafo acíclico. Os nós de entrada do grafo são os dados de *input*, os nós de saída são o resultado e os nós intermédios têm funções de activação, que definem o *output* desses nós, dado um conjunto de dados de entrada. Sendo que o grafo é acíclico, todos os nós podem ser organizados por camadas [23]. É possível verificar mais detalhadamente na Figura 2.3.

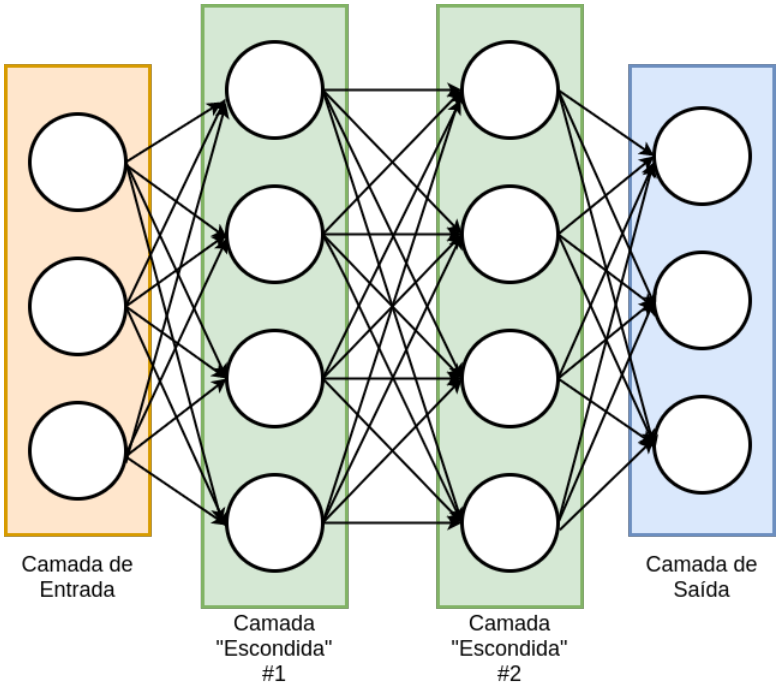

Figura 2.3: Exemplo visual de *Multilayer Perceptron*

Este tipo de algoritmo pode ser classificado como uma rede neuronal artificial *feed-forward*, sendo que os dados são passados de uma camada para a seguinte, sofrendo alterações, devido às funções de activação.

Para ser possível treinar de maneira mais eficiente a rede, foi implementado a técnica de *backpropagation*. A rede consegue ajustar os pesos de forma automática, calculando o gradiente do erro quadrático ou grau de ajuste dos pesos, entre cada treino do modelo.

Este ajuste não é certo, podendo o modelo tornar-se mais preciso ou não. Na execução seguinte, os pesos actualizam, podendo então, o modelo ficar mais preciso. [24]

Porém, também este tipo de redes têm limitações. Como cada nó de uma camada está ligado a todos os nós da camada seguinte, quanto mais escalamos a rede, maior o número de ligações, o que torna a rede computacionalmente cara, o que impede que seja escalável.

2.3.3 Redes Neuronais de Convolução

As Redes Neuronais de Convolução (CNN - Convolutional Neural Networks) são uma solução ao problema introduzido pelo MLP. O objectivo destes é diminuir o tamanho do modelo, retirando apenas a informação necessária para que o modelo possa tomar uma decisão. Esta diminuição é feita utilizando processo de convolução dos dados. As CNN são, normalmente, aplicadas a problemas onde seja necessário analisar imagens.

Uma CNN é repartida em 3 partes:

- Convolução;
- *Pooling*;
- *Fully Connected*.

Numa primeira fase são intercaladas várias camadas de Convolução e *Pooling*. Nas camadas de convolução, ou seja, no processo de operações entre matrizes, de modo a retirar as características da imagem de entrada, a convolução preserva a relação espacial entre pixeis, aprendendo as *features* da imagem, usando pequenos quadrados de dados de input.

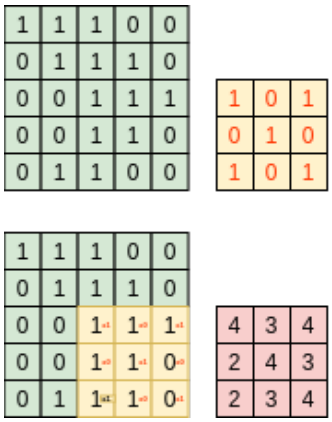

Figura 2.4: Exemplo visual de Convolução

Para entender melhor este processo, será utilizado o exemplo da figura 2.4, onde é possível verificar que a matriz amarela (o filtro), é passada sobre a matriz verde (a imagem original), começando no canto superior esquerdo e acabando no canto inferior direito. Por exemplo, quando a matriz amarela sobrepõe-se no canto inferior direito, ao multiplicar

as duas matrizes a soma dos valores é 4, tal como podemos ver na matriz vermelha na linha 3 e coluna 3. No final deste processo ficamos com a matriz vermelha, sendo esta denominada por *Feature Map*, tendo esta o mesmo tamanho da matriz de entrada. É de notar, que em cada fase de convolução existem mais do que uma matriz de filtros.

Após a fase de convolução, normalmente, é feita uma rectificação de valores. No caso da função de activação utilizada nesta dissertação, ReLU ou *Rectified Linear Unit*, a tarefa desta função é retirar os valores negativos, que resultaram da fase de convolução. O objectivo desta camada é aumentar as propriedades não lineares da rede, sem afectar os campos receptivos da camada de convolução.

A terceira camada é a camada de *pooling*. Esta camada está encarregue de criar uma amostragem mais pequena da matriz, de modo a reduzir o tamanho espacial da representação. Assim, é feito uma diminuição no número de parâmetros, o que permite diminuir a memória utilizada, e o tempo de computação. Esta amostragem é feita com base numa função. Uma função muito utilizada é o *MaxPool*. Esta função retorna o maior valor que se encontra numa sub-matriz da matriz original, exemplificado pela figura 2.5 que se segue.

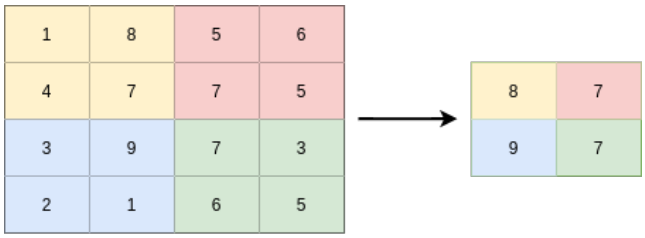

Figura 2.5: Exemplo visual de *MaxPool*

Por fim, o resultado de uma ou mais passagens por este conjuntos de blocos de processamento, é passado para uma secção da rede semelhante a um MLP. O processo todo de uma CNN está exemplificado na figura 2.6.

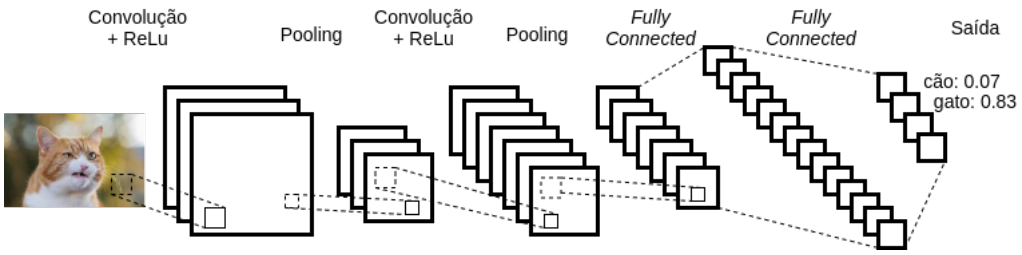

Figura 2.6: Exemplo visual de uma CNN

Uma imagem pode ser representada numa matriz, em que cada pixel corresponde a um elemento na matriz, na mesma posição do pixel na imagem, como se pode ver na figura 2.7. Neste caso, a cor da imagem está na escala dos cinzentos, ou seja, valores compreendidos entre o 0 e 255.

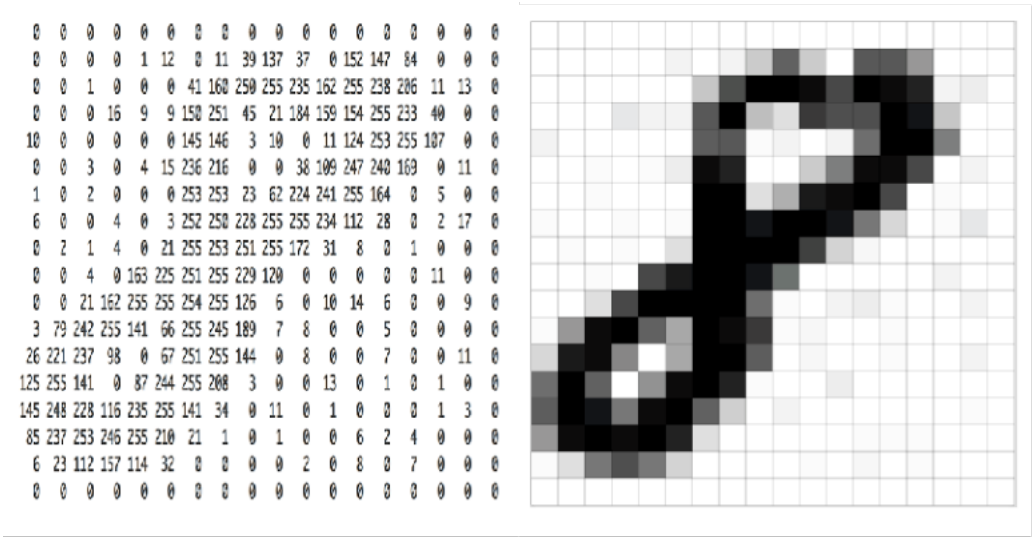

Figura 2.7: Representação de uma imagem num dataset

Uma imagem a cores contém mais que um valor por pixel. A cor de cada pixel pode ser descrita utilizando a escala RGB, que identifica as componentes vermelha (R - Red), verde (G - Green) e azul (B - Blue). Cada um destes canais é uma matriz de 2 dimensões, semelhante à das imagens na escala de cinza, mas contendo o valor do canal para cada pixel. Assim, as 3 matrizes de 2 dimensões ficam sobrepostas, e são analisadas as mesmas secções, ao mesmo tempo.

2.3.4 VGG

A arquitetura utilizada neste trabalho foi inspirada no modelo desenvolvido pelo *Visual Geometry Group* da Universidade de Oxford. Esta arquitectura foi concebida para o *ImageNet Challenge* 2014, e apresentou uma nova forma de estruturar as camadas de convolução para imagens de grandes dimensões, que até a esse ponto ainda não tinha sido utilizado.

O modelo desenvolvido utiliza um maior número de camadas de convolução, ou seja, uma maior profundidade, mas para compensar o maior número de camadas de convolução, diminuí o tamanho da janela de convolução (para uma janela 3×3). Com este trabalho, o grupo reforçou que, a performance das redes neuronais pode ser melhorado com o aumentar da profundidade das camadas, para problemas em que são introduzidas imagens de grande dimensão, que no caso deste desafio, as imagens tinham uma dimensão 224×224. Também é de notar que os modelos construídos a partir desta arquitectura são bons a generalizar para um grande número de tarefas e *datasets* [29].

2.3.5 Formato da Estrutura do Modelo

De modo a explicar a arquitectura do modelo utilizado para cada um dos testes, utilizou-se uma notação escrita. Existe 3 tipos de camadas para cada modelo: Convolução, *Pooling* e *Fully Connected*. Nesta notação não se tomou em conta uma camada de rectificação dos valores (ReLU), apesar desta ser utilizada sempre que é feita uma convolução.

A notação utilizada para as camadas de convolução foi a seguinte:

$$128\ C\ 3 \times 3 \tag{2.1}$$

128 indica o número de filtros utilizados na camada, enquanto 3×3 indica o tamanho de cada filtro.

Para as camadas de *Pooling*, a notação é a seguinte:

$$2 \times 2\ MP\ 2 \times 2 \tag{2.2}$$

O primeiro 2×2 indica o tamanho do filtro de pooling, enquanto o segundo 2×2 indica a passada (*stride*) que o filtro tomou sobre os dados. MP indica o tipo de *pooling* utilizado, sendo neste caso o *Max Pooling*. Na maioria dos casos será *Max Pooling*, caso contrário será indicado o tipo de *pooling* utilizado.

Por fim, nas camadas *Fully Connected* a notação é a seguinte:

$$1024\ FC\ 2 \tag{2.3}$$

1024 indica o número de nós que formam a camada anterior, ou seja o input para a camada atual, e o 2 indica o número de nós da camada atual, ou seja o output.

2.4 Ficheiro *Ubyte*

Para este projecto escolheu-se usar ficheiros *ubyte*. Estes ficheiros foram descritos por Yann Lecun no seu site, e foi utilizado para criar o famoso MNIST. Um *dataset* divide-se em dois ficheiros *ubyte* - um para os dados e outro para as etiquetas (ou *labels*).

O ficheiro de *labels* tem a seguinte estrutura:

Offset	Tipo	Valor	Descrição
0	Inteiro	0x00000801 (2049)	Número Mágico
4	Inteiro	# de etiquetas no *dataset*	-
8	*Byte*	0 ... 9	etiqueta
...	*Byte*	0 ... 9	etiqueta
xxx	*Byte*	0 ... 9	etiqueta

Tabela 2.1: Ficheiro *Ubyte* de Etiquetas

O ficheiro de dados tem a seguinte estrutura:

Offset	Tipo	Valor	Descrição
0	Inteiro	0x00000803 (2051)	Número Mágico
4	Inteiro	# de imagens no *dataset*	-
8	Inteiro	# de linhas por cada imagem	
12	Inteiro	# de colunas por cada imagem	
16	*Byte*	?	pixel
...	*Byte*	?	pixel
xxx	*Byte*	?	pixel

Tabela 2.2: Ficheiro *Ubyte* de Dados

O número mágico descreve o número de dimensões que cada dado tem e, é expresso em hexadecimal 0x00000803 (2051). O último número representa o número de dimensões de cada dado no *dataset*.

Voltando à figura 2.7 onde se trata de imagens a preto e branco, esta tem três dimensões - largura, altura e o valor de cada pixel. Cada pixel da imagem representa o valor na escala do cinzento (0 a 255), e deste modo conseguimos visualizar uma imagem. Se tratasse de uma imagem a cores seria necessário adicionar uma nova dimensão - o número de cores base, utilizadas pelo o modelo de cores RGB (*Red*, *Green*, *Blue* - tem três cores base, que ao junta-las é possível representar qualquer outra cor). Uma imagem a cores será, deste modo, repartida em 3, uma imagem para cada cor base do modelo de cores - uma imagem terá em cada pixel os valores dos vermelhos, outra imagem terá os valores dos verdes, e a terceira dos azuis. Ao juntar cada pixel da imagem ao longo da terceira dimensão, é possível obter a cor real. Neste caso o número mágico passa a ser 0x00000804 (2052).

2.5 Métodos de Comparação

2.5.1 Precisão

Em *machine learning*, a precisão pode ser definida como a percentagem de previsões que estavam realmente correctas, ou qual é a proporção de identificações positivas que estavam correctas. A precisão utilizada nesta dissertação foi a seguinte:

$$Precisão\ (\%) = \frac{TP}{TP + FP} \times 100\% \tag{2.4}$$

TP é o número de positivos verdadeiros (*True Positive*), ou seja, o número de dados positivos que foram identificados como tal. Enquanto FP (falsos positivos), é o número de negativos que foram identificados como positivos.

2.5.2 *Recall*

O *Recall* pode ser definido como a quantidade de positivos que foi correctamente identificado. O *Recall* pode ser calculado deste modo:

$$Recall\ (\%) = \frac{TP}{TP + FN} \times 100\% \tag{2.5}$$

A diferença entre a precisão e o *recall* é que o *recall* toma em consideração todos os positivos disponíveis, enquanto a precisão toma apenas em consideração os positivos identificados.

2.5.3 *F-Score*

O *F-Score* combina a precisão e o *recall*, e é um método de cálcular a precisão de um modelo sobre um dataset.

$$\textit{F-Score}\ (\%) = 2 \times \frac{precisão * \textit{recall}}{precisão + recall} \times 100\% = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \times 100\% \tag{2.6}$$

Deste modo é possível ver que o *f-score* dá igual importância à precisão e ao *recall*, sem tomar em consideração os TN.

Este foi o método escolhido para comparar modelos neste trabalho, mas outros métodos podem ser utilizados, como o Coeficiente de Correlação de Matthews ou o Kappa de Cohen.

Capítulo 3

Análise e escolha do *Dataset*

3.1 AMIGOS

Antes de procedermos ao treino de modelos, foi necessário analisar qual seria o *dataset* existente que melhor conseguiria cumprir o objectivo desta dissertação.

O primeiro *dataset* existente testado neste projecto foi o AMIGOS. Este foi construído a partir de uma experiência onde cada participante viu um excerto de um filme. Enquanto estão a ver o excerto do filme, estão ligados a sensores de EEG (Electroencefalograma), ECG (Eletrocardiograma), e a uma de GSR (*Galvanic Skin Response*), e também estão a ser filmados. Após verem o excerto, os participantes avaliaram o que foi visto, utilizando níveis afectivos (valência e *arousal*). Para diferenciar os participantes, foi feito um questionário de personalidade, de modo a entender previamente quais são os traços da sua personalidade.

O *dataset* AMIGOS contém um grande conjunto de dados, composto por dados de diferentes dimensões, para 30 participantes. Entre estes estão os dados biométricos, que entre eles capturam diferentes dimensões das reacções dos participantes. Além disso, cada participante foi gravado, com recurso a uma câmara, para analisar as expressões faciais em relação aos vídeos que estão a ver.

Este *dataset* divide-se em duas partes. A primeira é composta por 40 participantes, que viram 16 vídeos de curta duração – com cerca de um minuto. A segunda é composta por 37 participantes que viram 4 vídeos de longa duração. Ao todo, existem 788 imagens, e as suas correspondentes etiquetas.

3.1.1 Limitações

Este *dataset* tem duas limitações. A primeira é o número de excertos. 778 imagens são muito poucas, e é bastante provável que a performance do modelo não seja satisfatória. A segunda é o problema de conjugar os diferentes tipos de ficheiros, sendo que existe um para cada uma dos sensores, mais o questionário de personalidade. Isto pode levar a uma arquitectura demasiado complexa. Sendo que esse não é o objectivo deste trabalho,

e o *dataset* não demonstra ter dados suficientes, foi decidido procurar outros *datasets* que melhor enquadrem no objectivo do trabalho.

3.2 DEAM

DEAM ou *Database for Emotional Analysis of Music*, é um *dataset* que tem como objectivo identificar as emoções em músicas. O DEAM é composto por 1802 excertos e músicas inteiras, tendo os valores de valência e *arousal* por cada 500 ms e uma etiqueta sobre o som inteiro.

Este *dataset* é composto por três partes. As músicas com nomes entre 1 e 1000 foram inseridas no *dataset* em 2013, estas incluem meta-dados sobre a hora da audição e a disposição da pessoa. Na segunda parte incluíram-se as músicas com nomes entre 1000 e 2000, mais os meta-dados sobre a confiança da anotação, familiaridade com a música, gosto da música, entre outros dados relevantes. Na última parte, incluíram-se os valores entre 2000 e 2058, onde as músicas teêm uma maior duração, e meta-dados sobre o gosto pessoal da música.

No final de ouvir as músicas, foi pedido aos participantes que preenchessem um formulário sobre a música, no qual foi-lhes pedido para identificar o valor de Valência e de *Arousal* para a música. Este tipo de dados iremos chamar de estáticos, pois são fixos no tempo.

Como o *dataset* inclui dados dinâmicos e estáticos, as anotações foram divididas pelos dois, tendo as anotações dinâmicas valores do valência e de *arousal* a cada 500 milissegundos, para cada pessoa que ouviu a música; os valores médios das anotações dinâmicas, ou seja, a média, por cada 500 ms, de todas as pessoas que ouviram a música; e os valores estáticos. Existe também a média e o desvio padrão das anotações estáticas por música.

3.2.1 Análise detalhada

Ao contrário do *dataset* anterior, para identificar se este seria utilizado para este trabalho, foi preciso analisa-lo melhor. Para isso, o passo foi entender se os valores dinâmicos transmitem informação relevante. Assim, foram criados gráficos para visualizar os valores da valência e *arousal* por música. No gráfico seguinte, cada pessoa que ouviu a música é identificada por uma cor diferente.

Na figura 3.1, a linha a tracejado representa o valor médio das 10 pessoas que ouviram a música. Como é possível observar, estas têm reacções bastante diferentes ao ouvir a mesma música e não é possível ver uma semelhança entre os 10. Pode ser interessante fazer a divisão pelos 4 quadrantes, podendo haver semelhança, entre as pessoas desse quadrante, ou seja, se uma pessoa, neste caso, se situa no 4º quadrante (Valência positiva, mas Arousal negativo) pode ser constante com a outra pessoa que também se situa no 4º quadrante, para todas as músicas.

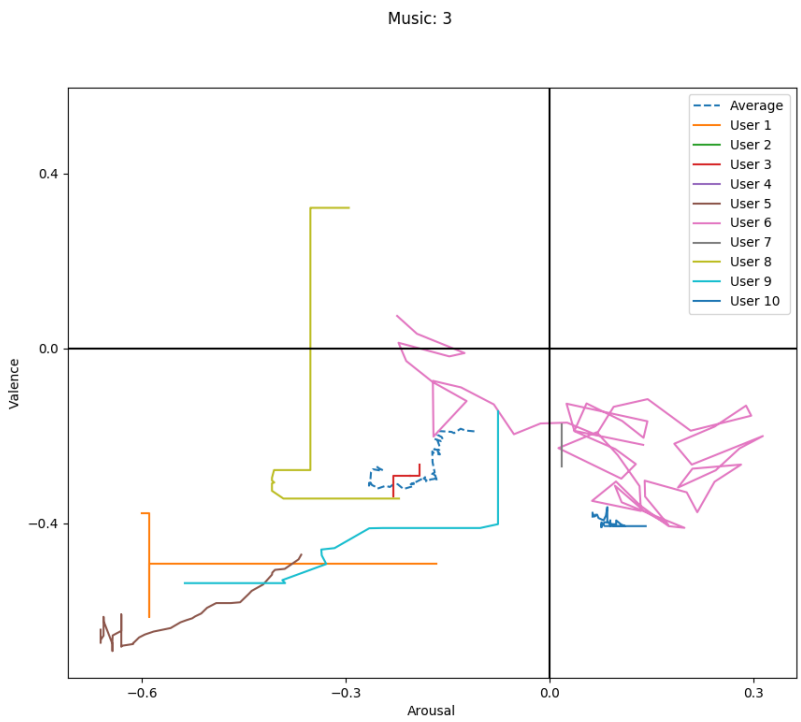

Figura 3.1: Distribuição do VA no DEAM

Com esta imagem é possível ver que a criação de uma interpolação de valores, para os pontos em falta, não seria fácil de produzir.

3.2.2 Limitações

Como é possível ver na distribuição do VA, ao longo do tempo não é possível prever como é que uma pessoa reage a uma música, o que leva a duas limitações: primeira o que deverá ser feito em caso de falta de dados, e segunda como deverá ser feita a analise dos resultados.

Para a primeira limitação, a solução não é linear e teria que ser analisado caso a caso. Dado que isto não é viável, e que mesmo que fosse analisado caso a caso, não é certo que o resultado fosse o correto. Se a solução passasse por não utilizar esse dado, poderia levar a uma redução massiva de dados. A segunda limitação é também devido a imprevisibilidade das anotações. Como não é certo a evolução de um dado, não existe maneira de analisar os resultados do modelo. Por estas razões foi decidido procurar outros *datasets* que não tenham estas limitações.

3.3 EMOMUSIC

EMOMUSIC é um *dataset* criado para a análise emocional de música. É composto por 100 músicas de diversos géneros e de diversos artistas. Os géneros de música presentes neste *dataset* são o *Blues, Classical, Country, Electronic, Folk, Jazz, Pop e Rock*, e têm a seguinte distribuição:

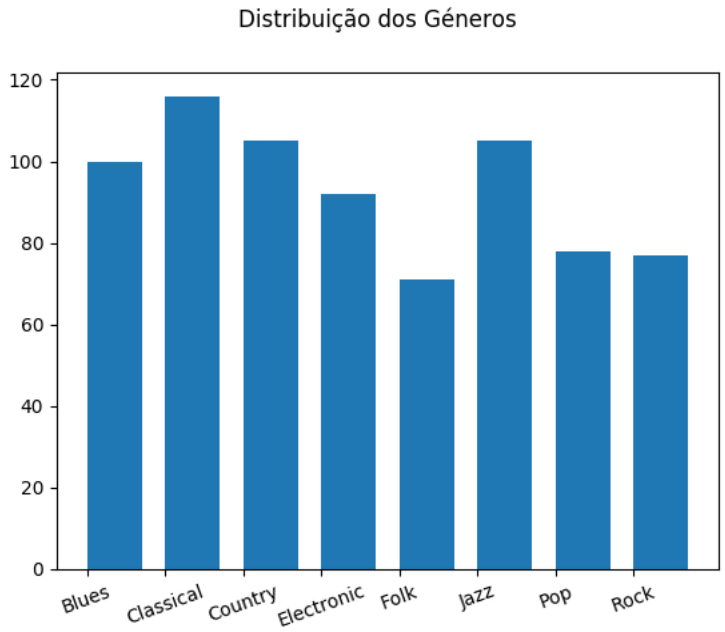

Figura 3.2: Distribuição das músicas no EMOMUSIC

Para a colecção de dados, os autores utilizaram o *Mechanical Turk*, uma plataforma disponibilizada pela Amazon, onde pessoas são pagas para fazer certas tarefas, que não podem ou nem deviam ser feitas automaticamente por um computador. O objectivo era classificar o VA de cada música que ouviram.

Para que não houvesse problemas com a qualidade dos dados, como no DEAM, os autores implementaram um processo de selecção com várias camadas, de modo a que, pudessem identificar quem é que tinha entendido completamente a tarefa a executar. Das 1000 pessoas iniciais, 778 completaram a primeira tarefa de qualificação. Dessas 778, apenas 287 foram convidadas para a primeira tarefa de classificação. Dessas 287 pessoas, apenas 100 completaram no mínimo uma tarefa de classificação. Dessas 100 pessoas 57 eram homens e 43 mulheres, com idades médias de 31.7 anos, sendo que 72% estavam situados nos Estados Unidos da América, 18% na Índia, e 10% espalhados pelo resto do mundo.

A anotação do VA foi separada para cada *clip*, sendo que as pessoas ouviam a música duas vezes, na primeira vez anotavam para valência e na segunda para *arousal*. Cada pessoa ficou, em média, 7 minutos e 40 segundos a anotar clipes de 45 segundos, tanto

para valência como para arousal. Cada pessoa anotou, em média, 107.9 músicas, No total foram recolhidas mais de 20 000 anotações.

Para este *dataset* foram recolhidos dois tipos de anotações: dinâmicas e estáticas. As anotações dinâmicas são os valores de VA que são recolhidos cada 500 ms, e têm valores entre -1 e 1. As anotações estáticas são os valores do VA que são recolhidos no final da música, onde cada trabalhador dá uma classificação tanto para a valência e para o *arousal*, no intervalo entre 1 e 9.

Para recolher as anotações dinâmicas, os autores criaram uma interface de anotações de músicas online, como pode ser visto na Figura 3.3

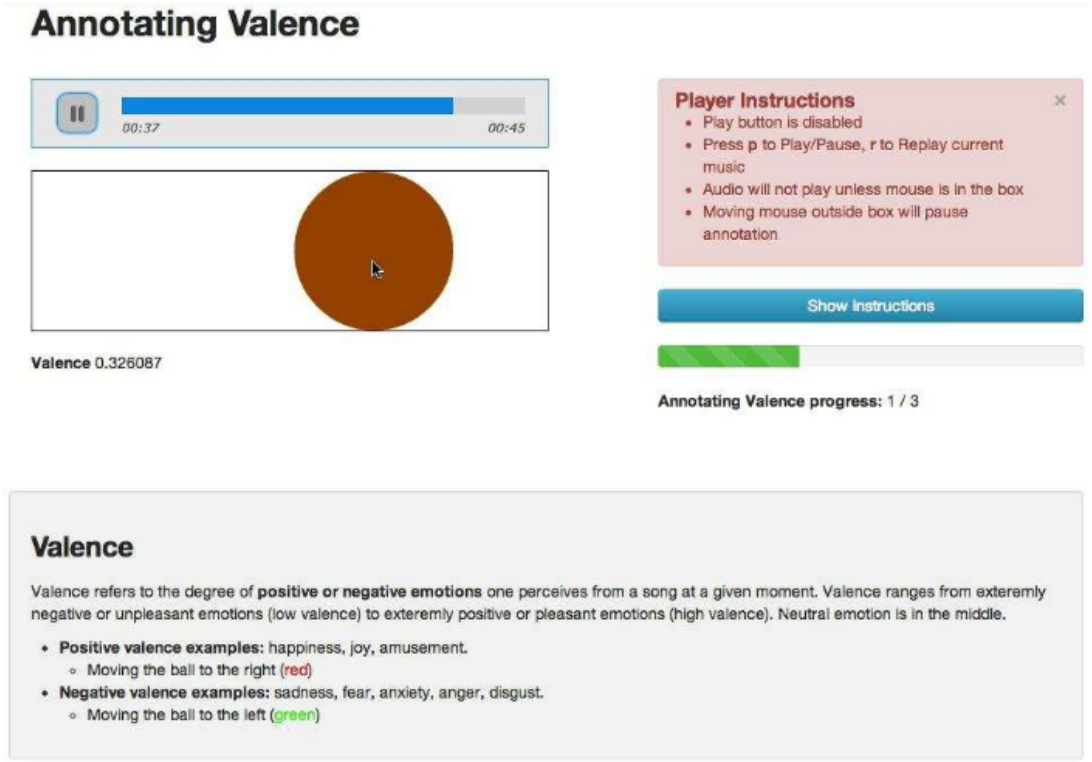

Figura 3.3: Interface de Anotação de Músicas no EMOMUSIC

3.3.1 Análise detalhada

Um ponto que deve ser verificado é a distribuição dos valores de VA das anotações dinâmicas. Primeiro verificou-se que a distribuição no intervalo de valores original, ou seja, entre 1 e 9. Como é possível ver na figura 3.4 foi introduzido no gráfico os eixos de referência, que neste caso situam-se em 4.5 de valência e 4.5 de *arousal*.

Durante a música, os trabalhadores deslizam a bola para indicar o VA nesse momento. De modo a maximizar a ocupação dos trabalhadores, cada vez que retiravam o cursor do rato da caixa rectangular, a música parava. A frequência de amostragem depende do *browser* que os trabalhadores utilizavam. O intervalo médio utilizado foi de 0.23 segundos

(4.3 Hz). Para igualar todos os resultados, os valores foram passados para uma frequência de 1 Hz.

Os primeiros 5 segundos das anotações dinâmicas foram descartados devido à instabilidade nos seus valores. Os valores de VA dinâmicos para cada música foram gerados através da média de todos os trabalhadores que anotaram a música, de modo a obter uma verdade base.

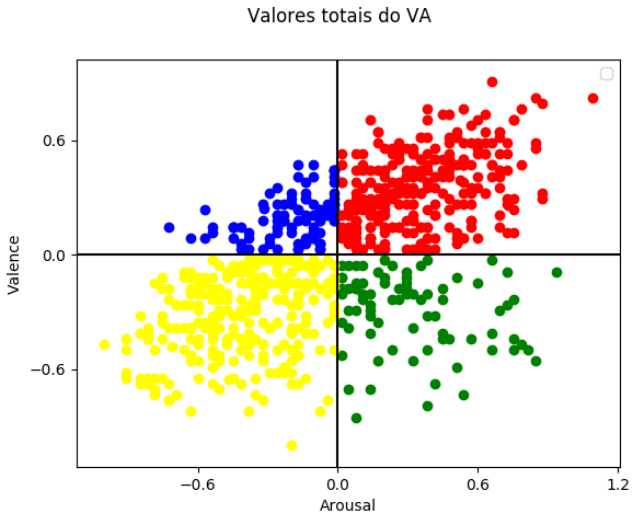

Figura 3.4: Valores Dinâmicos do VA no EMOMUSIC (utilizando a escala entre -1 e 1)

Para cada um dos quadrantes foi dada uma cor. No caso dos pontos que se situarem na linha entre dois quadrantes, a cor resultante é da junção das duas cores. Da análise da figura 3.4 é possível ver que existem dois quadrantes com menos músicas - o segundo e o quarto, e que, a maioria dos valores situa-se no primeiro quadrante.

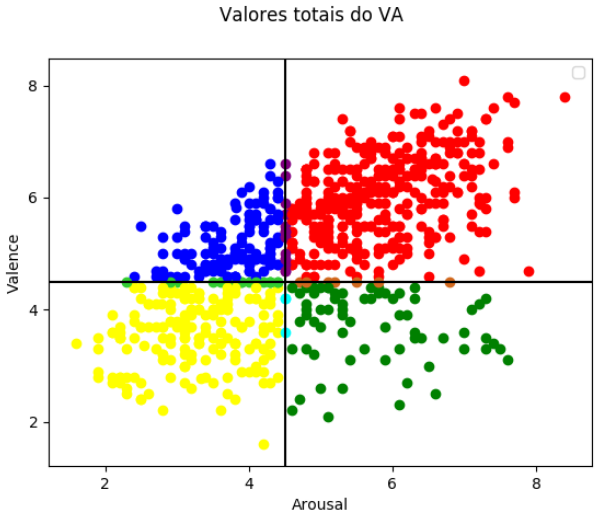

Figura 3.5: Valores estáticos do VA no EMOMUSIC (utilizando a escala de 1 e 9)

O mesmo foi feito com os valores estáticos, no entanto estes apresentam valores entre -1 e 1, como é possível observar na figura 3.5.

Foi feita a interpolação dos valores dinâmicos para o intervalo entre -1 e 1, de modo a poder comparar com os valores estáticos, o gráfico resultante é uma tradução directa do primeiro gráfico.

A seguir, são comparados os valores dinâmicos com os estáticos. É esperado que haja diferenças, sendo que são duas anotações feitas de maneiras diferentes, em alturas diferentes. Mas também, é esperado que, haja alguma semelhança, ou seja, que se encontrem relativamente perto, ou no mínimo no mesmo quadrante. Isso dirá a veracidade das respostas dos trabalhadores.

O gráfico 3.6 compara o ponto interpolado dos valores estáticos, com a média dos valores dinâmicos.

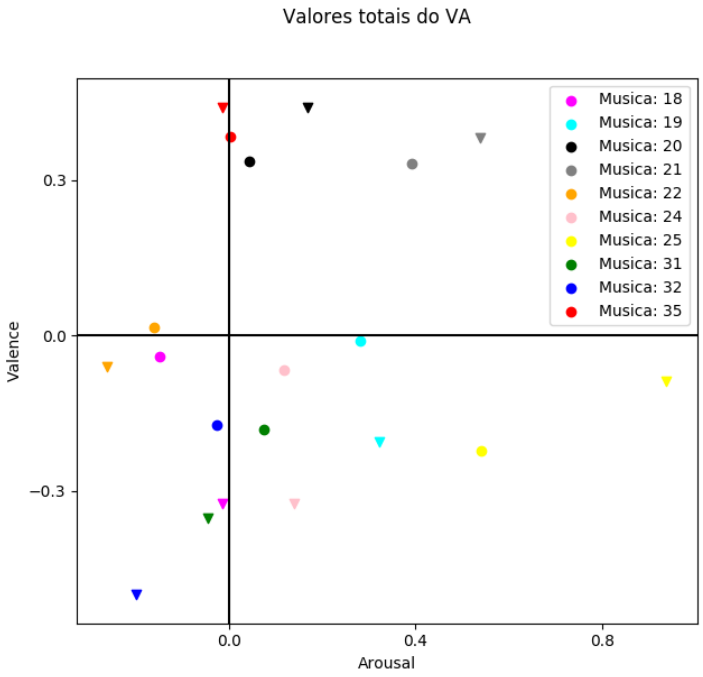

Figura 3.6: Diferença entre Valores Dinâmicos e Estáticos

Cada música é representada com uma cor diferente. As anotações dinâmicas estão representadas com uma bola, enquanto as estáticas estão com um triângulo. Como é possível observar, em quase todos os pontos a diferença ou é pequena, ou no mínimo situam-se no mesmo quadrante. É possível afirmar que os dados são fidedignos, ou seja, que os trabalhadores fizeram a tarefa de maneira correta.

3.3.2 Limitações

Sendo que não foi possível identificar outros trabalhos que utilizem este *dataset*, não existe maneira de confirmar que os resultados obtidos são bons, visto que, a valência e

arousal são subjectivos, não havendo coerência entre os diferentes participantes. Outro problema é a falta de *datasets* semelhantes, para, caso seja necessário, poder adicionar mais dados ao treino do modelo. Dado estes problemas, decidimos avançar para um *dataset* que fosse mais simples de validar, e em que, a classificação dos dados utilizados possa ser facilmente verificada.

3.4 AudioSet

Devido à dificuldade de encontrar *datasets* que tenham um número de publicações, que fossem considerados suficientes para uma comparação com o trabalho realizado, foi feito o pivô para um *dataset* baseado em emoções induzidas, ou seja, utilizando fontes de áudio que têm como o objectivo induzir uma determinada emoção. De modo a angariar um número considerável de áudios distintos, foi utilizado o Youtube como fonte.

O *dataset* AudioSet, produzido pela Google, é um *dataset* de grande escala, com cerca de 2 milhões de vídeos anotados manualmente. Esses 2 milhões de vídeos estão divididos em 527 classes, e de cada vídeo apenas é classificado um segmento de 10 segundos. Cada vídeo pode ter mais que uma classe, e cada classe tem uma qualidade estimada. A qualidade é calculada analisando 10 segmentos aleatórios de cada classe, e verificando se as classes estão corretas. [5]

Este *dataset* foi criado sobre uma ontologia. Uma ontologia é um método de modelação de dados, que representa um conjunto de conceitos dentro de um domínio. Neste caso, existem 7 categorias: Sons Humanos, Sons Animais, Música, Sons de Coisas, Sons Naturais, Sons Ambiente, Sons Ambíguos.

Cada uma dessas categorias têm uma árvore de categorias, e cada uma das categorias pode ter outras categorias. O número de vídeos por cada categoria não é igual, e algumas categorias têm milhões de vídeos.

3.4.1 Escolha de AudioSet

Ao utilizar este *dataset* podemos ter com uma determinada certeza (qualidade da classe), que o vídeo presente na classe tem de facto o atributo esperado. Isto deve-se ao facto da classificação de um certo vídeo ser confirmado por mais do que uma pessoa.

Outra razão é a quantidade dos dados. O AudioSet tem cerca de 2 milhões de vídeos, dos quais retiramos troços de 10 segundos. Apesar de não estarem repartidos igualmente por cada classe, continua a ser um número significativo de vídeos por classe.

Por fim, o facto das etiquetas serem simples. Isto permite-nos aumentar os dados, utilizando por exemplo, a rotação de dados, e manter a mesma qualidade. Como será visto mais a frente, esta táctica foi utilizada várias vezes.

Capítulo 4

Implementação

4.1 Arquitectura do Modelo

A arquitectura utilizada nesta dissertação é baseada no modelo VGG [29]. Como a arquitectura descrita pelo mesmo, não funcionaria devido ao tamanho dos dados de entrada para o problema estudado, foram testadas várias variantes da arquitectura base, e seleccionada a que obteve melhores resultados. Entre as variantes estudadas, é de salientar que o algoritmo de cálculo do gradiente escolhido foi o *Adam-Optimizer*, por ser um dos mais utilizados, e o que ofereceu melhores resultados[10]. Este algoritmo é uma extensão dos algoritmos de cálculo de gradiente estocásticos, nos quais é mantido uma única taxa de aprendizagem. O *Adam-Optimizer* calcula taxas individuais e adaptativas para os diferentes parâmetros.

Para a construção deste modelo foi utilizado o *Tensorflow*, uma biblioteca de *machine-learning* focada no treino e inferência de Redes Neuronais. Cada treino foi feito com 100 iterações, ou seja, 100 passagens pelo *dataset*, em que o *batch size* é 32, e o *learning rate* é 0.00010.

A arquitectura deste modelo é a seguinte:

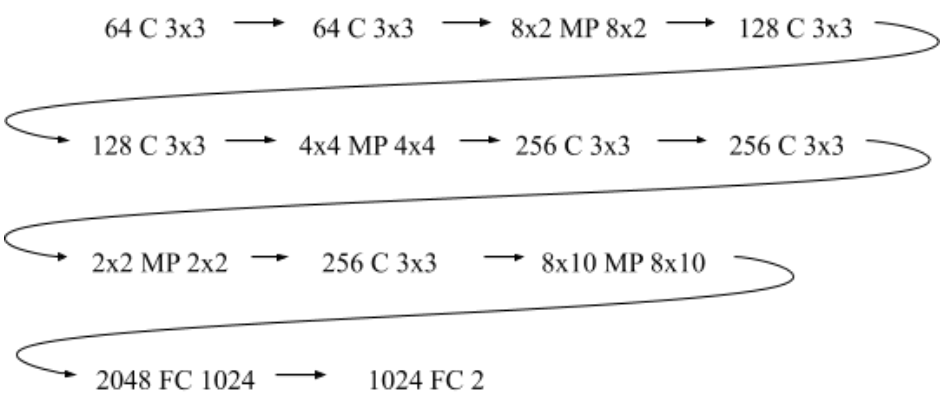

Figura 4.1: Arquitectura da Rede Neuronal Desenvolvida

Os dados utilizados para treinar o este modelo são consideravelmente maiores que as utilizadas pelo VGG, com a dimensão 512×320, em que o conteúdo é o espectograma de um troço de áudio do *dataset*. Cada espectograma tem 512 frequências e dura 10 segundos, sendo que em cada segundo são retiradas 32 amostras, o que equivale a 320 amostras totais.

O *output* do modelo é um *array* com tantas posições quanto classes disponíveis. Esse *array* é composto de zeros em todas as posições menos a posição que equivale à classe escolhida. No final é traduzido a posição para a classe.

4.2 Método de Criação do Dataset

A tabela 4.1 representa o *dataset* utilizado, que é um ficheiro CSV com 4 colunas. A primeira coluna indica o ID do vídeo no Youtube, a segunda e a terceira colunas indicam o ponto de inicio e fim do segmento (em segundos) em relação ao tamanho total do vídeo, e a quarta coluna contém as etiquetas que identificam o vídeo.

ID	Segundo de Começo	Segundo de Fim	*Labels*
–PJHxphWEs	30	40	/m/09x0r,/t/dd00088
–ZhevVpy1s	50	60	/m/012xff

Tabela 4.1: Exemplo do ficheiro de dados do AudioSet

Estas etiquetas são codificadas, sendo utilizado um outro ficheiro CSV para fazer a conversão. Uma amostra deste ficheiro pode ser visto na figura 4.2 existem três colunas, a primeira com o índex da etiqueta, a segunda com o ID, e a última com a classe por extenso.

Index	*Ontology Name*	*Display Name*
0	/m/09x0r	*Speech*
1	/m/05zppz	*Male speech, man speaking*

Tabela 4.2: Exemplo de ficheiro de *Labels* do AudioSet

Para facilitar o acesso aos dados, foi criado uma amostra do *dataset*, que contém apenas as classes que pretendemos utilizar. De forma a garantir o correcto funcionamento do modelo, garantiu-se que os vídeos com classe negativa não tenham também a classe positiva. Assim, foi criado um novo ficheiro JSON, que indica os as classes positivas, as classes negativas, os IDs dos vídeos dos quais será feito o *download*, e o inicio e fim do troço, para que possa posteriormente ser cortado do vídeo original. Este método permite-nos duas coisas: primeira, replicar resultados do *download* dos vídeos, assegurando que serão sempre os mesmos descarregados para cada classe, e segunda, permite cortar o

tempo de execução do programa, sendo que a parte de leitura dos ficheiros CSV do *dataset* é extensa, devido ao seu tamanho.

Para guardar os ficheiros áudio foi criada uma directoria-mãe, composta de uma directoria-filha por cada classe existente. Assim, cada directoria tem o nome da classe. Isto foi feito para melhorar a velocidade com que são produzidas as estatísticas dos *datasets*.

Após a criação do ficheiro JSON, é feito o *download* dos vídeos de cada classe, e são guardados numa directoria.

Para o *download* do vídeo foi utilizado a ferramenta YouTubeDL [1]. Esta ferramenta descarrega o vídeo, retira o áudio, e faz a conversão para WAV. Após o *download*, é feito um *downsampling* do áudio, passando de 42 KHz para 16 KHz. Este *downsampling* é realizado de modo a reduzir o tamanho dos dados. É também feito o corte do vídeo original, para o troço esperado.

Para isso, utilizou-se a ferramenta SOX [2]. Esta permite a conversão de áudio, o *downsampling* de um áudio, o corte de um áudio e muito mais. Como o SOX não consegue fazer a modificação do ficheiro actual directamente, é necessário criar um novo ficheiro com cada modificação do original. Para diminuir o espaço ocupado em disco, assim que uma modificação é feita, o áudio anterior é apagado. A seguir é verificado se o áudio tem mais ou menos que 10 segundos. Se tiver mais é feito o corte no final do troço, se tiver menos, é extendido. Neste caso, calculamos quanto tempo resta para fazer os 10 segundos, dividimos esse valor por 2, e o resultado é retirado do inicio e do final do segmento e é colado em duplicado.

Figura 4.2: Exemplo de uma extensão de um troço de áudio

No exemplo da figura 4.2 , é possível verificar que a laranja estão os troços do início do troço e a vermelho os troços de fim do segmento.

No final, cada áudio é colocado numa directoria com o nome da classe a que pertencem.

Para criar os *datasets* é necessário converter cada áudio num espectograma, e em seguida inseri-los num ficheiro *Uyte*. Para a conversão em espectograma foi utilizado a biblioteca *Python Scikit*. Para a criação do *Ubyte* foi criado uma biblioteca em *Python*.

Cada *dataset* criado é testado, para que seja detectado qualquer irregularidade com a sua criação.

[1] https://youtube-dl.org/
[2] https://bit.ly/3AChqcy

4.2.1 *Datasets* Abertos e Fechados

Um ponto essencial a ser estudado neste trabalho são os *datasets* ”abertos”. Neste caso, um *dataset* aberto é um *dataset* sujeito a mudanças, isto é, o que vai ser estudado é a evolução do mesmo modelo sobre diferentes versões do *dataset*. Ao longo das experiências o *dataset* irá mudar em relação às *labels* presentes em cada uma das classes. Com isto, é suposto simular o mundo físico devido a irregularidade das classes ao longo das experiências. A criação dos *datasets* será assente no *dataset* base (AudioSet), de modo a que, os resultados sejam comparáveis. Em cada experiência é feito uma previsão e uma explicação dos resultados obtidos.

Com *datasets* fechados o treino é sempre feito com o mesmo grupo de dados, o que normalmente leva aos melhores resultados no treino e na validação do modelo. Este tipo de *datasets* traz vantagens em situações em que o este consegue modelar um grande conjunto da tarefa. Sendo que alguns domínios não permitem encapsular a tarefa num *dataset*, devido ao seu tamanho ou complexidade, podemos utilizar *datasets* abertos. Desde modo, é possível criar vários *datasets* que agrupem partes do domínio a ser estudado.

4.2.2 Aumentação do *Dataset*

Um dos possíveis problemas encontrados é a falta de dados. É de esperar que, quando um modelo é treinado com mais dados seja melhor que um treinado com menos. Dado que não existem mais *datasets* uma solução possível é aumentação de dados.

Aumentação de dados refere-se ao conjunto de técnicas que permite criar dados novos, a partir de dados existentes. Estes novos dados são criados manipulando os dados existentes, desta maneira alterando o seu aspecto [19]. Sendo que o *dataset* utilizado trata-se de sons, vai ser estudado duas técnicas para o aumento de dados: rotação de dados e controlo de volume.

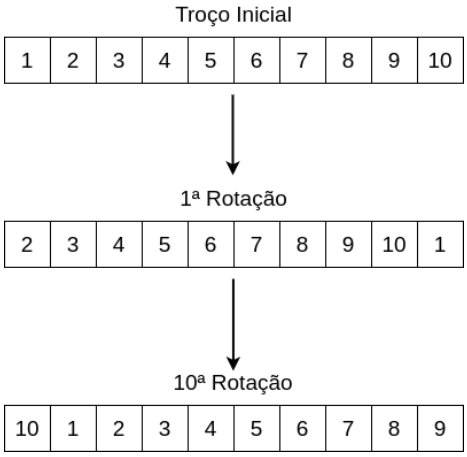

Figura 4.3: Exemplo de rotação de um troço de áudio

Na rotação de dados, cada segundo do vídeo é passado para o final do troço. Assim,

na primeira rotação, o primeiro segundo do vídeo é passado para o final. Na segunda rotação, o segundo segundo do troço é passado para o final, ficando a seguir ao primeiro troço, que tinha sido passado na rotação anterior. Ao todo, são feitas 10 rotações, sendo que, no inicio cada troço tem 10 segundos, e acaba quando o último segundo do troço inicial, que se encontra no inicio. Neste caso, é possível multiplicar por 10 o número de dados positivos. Mais detalhadamente, ilustrado na figura 4.3.

O espectograma final fica com o seguinte aspecto, como é possível ver nas figura 4.4, 4.5 e 4.6.

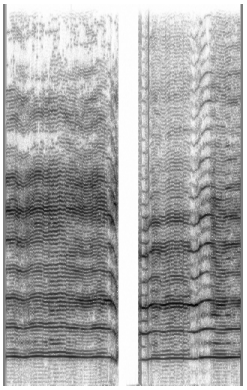

Figura 4.4: Rotação 0

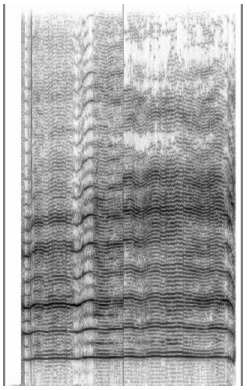

Figura 4.5: Rotação 5

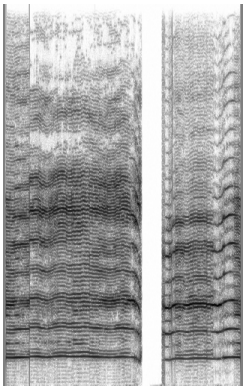

Figura 4.6: Rotação 10

Com este tipo de aumento de dados é possível o ver o efeito causado. Este método pode ter variação suficiente entre dados aumentados com base no mesmo troço, dado que as características do espectograma mudam de sítio entre rotações. Na figura 4.4 verifica-se uma secção no meio completamente branca, nesse momento não há som no áudio. Esta figura ilustra o áudio base, isto é, o áudio que vai ser rodado. Na figura 4.5 a secção branca encontra-se no final do espectograma, podendo verificar-se então o efeito da rotação de dados. A rotação encontra-se a meio, ou seja, na quinta posição. Para terminar, na figura 4.6 é possível ver a posição final da rotação de dados, em que o segmento do primeiro segundo encontra-se no fim. Nesta rotação, o secção a branco encontra-se na posição imediatamente anterior na qual começou.

No controlo de volume, alteramos o volume do troço inicial. A ferramenta SOX permite aumentar ou diminuir o volume do troço original, criando um novo troço, mas com o volume alterado. Neste caso, existe uma possível limitação, ao aumentar ou diminuir demasiado o volume, as características do troço podem-se perder ou tornar imperceptíveis.

Analisando as figuras, verifica-se que o processo de controlo de volume parece introduzir pequenas alterações ao troço original. Como é possível ver na figura 4.9, no troço que foi criado aumentando o volume para 120% do original, parece introduzir ruído no canto inferior direito do espectograma. O facto de introduzirmos ruído pode ser uma ajuda no treino do modelo.

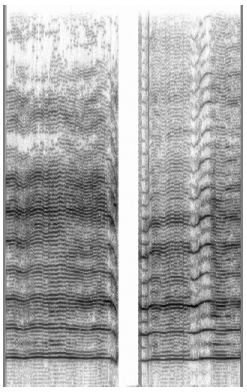

Figura 4.7: 80% Volume

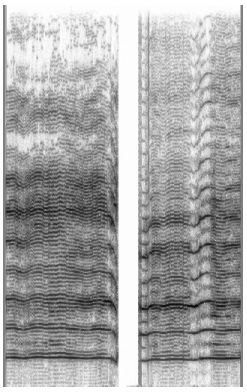

Figura 4.8: Volume Orig.

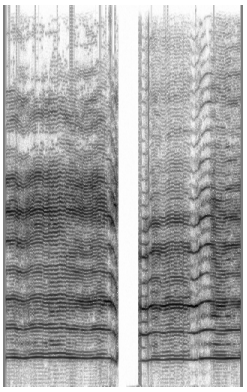

Figura 4.9: 120% Volume

4.3 Teste do modelo utilizando um filme

Para testar os modelos, é passado um filme para identificar se o modelo permite classificar correctamente os segmentos do filme que são positivos. Para o teste foi utilizado o filme *Saw*(2004). Após o *download* do filme, foi feito o mesmo processo usado para os vídeos de Youtube: retirou-se o áudio e realizou-se o *downsampling*. De seguida, o áudio do filme foi repartido em segmentos de 10 segundos, possibilitando cada segmento ter a mesma dimensão dos vídeos de Youtube. Esta segmentação foi feita com sobreposição dos troços a cada 5 segundos:

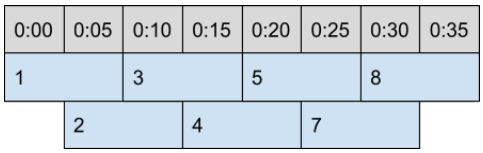

Figura 4.10: Sobreposição dos segmentos do filme

Para confirmar os resultados do teste, foi feito um ficheiro de *labels* para o filme. A primeira coluna indica a ordem do segmento, e cada uma das colunas seguintes tem uma das *labels* escolhidas do AudioSet.

segmento	*screaming*	*yell*	*gunshot*	...	*other*
0	0	0	0	...	1
1	0	0	0	...	1
2	0	0	0	...	1

Tabela 4.3: Tabela de composição do filme

Esta classificação dos segmentos do filme foi feita manualmente. Cada segmento foi ouvido, e atribuído uma etiqueta correspondente. Os segmentos foram apenas classificados com as *labels* que seriam usadas na construção de *dataset*, para o treino do modelo. A etiqueta *Other* identifica segmentos em que não é possível classificar com qualquer uma das outras etiquetas.

4.3.1 Composição do Filme

O filme foi dividido em 1232 segmentos. A cada um dos segmentos foi atribuído uma ou mais das seguintes etiquetas.

Etiqueta	Número Segmentos
Screaming	132
Yell	116
Gunshot	18
Music	72
Speech	515
Singing	8
Camera	32
Run	0
Silence	9
Laughter	1
Other	354

Tabela 4.4: Número de Segmentos por cada Label

Estas etiquetas serão utilizadas para analisar os falsos positivos classificados pelo modelo. O modelo apenas classifica positivo ou negativo, mas como a classe negativa é composta por mais do que uma etiqueta, é necessário perceber em que etiquetas o modelo tem mais dificuldade em classificar correctamente.

4.4 Denominação de *Datasets* e Experiências

Os *datasets* e experiências criadas têm a sua própria denominação. Os *datasets* tem a denominação *Vx*, em que o *x* é substituído pela sua versão. Esta denominação é uma abreviação do nome que é atribuído na sua criação: *audioset_train_data_vx-ubyte.gz*. Cada dataset é repartido em quatro: dados de treino (*audioset_train_data_vx-ubyte.gz*), etiquetas de treino (*audioset_train_label_vx-ubyte.gz*), dados de validação (*audioset_test_data_vx-ubyte.gz*) e etiquetas de validação (*audioset_test_label_vx-ubyte.gz*).

As experiências por sua vez têm a denominação de *YT0XX*. YT é a abreviatura de *Youtube*, a fonte de vídeos utilizados para a criação do dataset. O *XX* indica a versão do dataset.

Capítulo 5

Resultados e Discussão

5.1 *Dataset* V1

A primeira experiência foi feita com uma *dataset* com a seguinte constituição:

Nome	Classes Positivas		Classes Negativas		Total
	Classes	# de Troços	Classes	# de Troços	
V1	*Screaming*	758	*Speech*, *Singing*, *Yell*, *Run*	756 (189 por classe)	1514

Tabela 5.1: *Dataset* V1

As classes escolhidas são aquelas que esperamos que um filme de terror tenha. O foco nesta fase é criar uma *baseline* de comparação com os restantes *datasets* e com as diferentes variações no método de construção do mesmo.

Neste ponto serão estudados dois métodos. O primeiro é a gravação do modelo no final do treino, ou seja, o modelo gravado é sempre o da última iteração, sendo ele o melhor modelo do treino ou não (experiência YT001). O segundo método é gravar o modelo se a precisão do mesmo for melhor do que o último modelo gravado. O modelo em memória continua o treino, e pode melhorar ou piorar. Este segundo método tem o nome de *Save-Best* (experiência YT002). A suposição é que poderemos ter melhores resultados se apenas utilizarmos as melhores versões do modelo no teste.

Por fim, irá também ser estudado se um treino com validação do modelo, produz melhores resultados, do que, um treino sem validação (experiência YT003). Dado que, temos um número limitado de dados, a divisão do *dataset* de treino em dois (um para o treino e um para a validação), poderá produzir piores resultados, do que, utilizar o *dataset* todo para o treino. Ao mesmo tempo, o facto de não ser feito a validação, pode levar a uma pior afinação dos parâmetros (ou pesos) do modelo, conduzindo a *overfitting*, o que leva a piores resultados, pois o modelo mostra-se ineficaz a prever novos dados.

5.1.1 Avaliação da Experiência YT001

Esta experiência serve para a produção do *baseline*. Para esta experiência, o modelo utilizado foi de gravar sempre a última iteracção.

ID	Treino	Validação	Teste (Filme)		
	Precisão		Precisão	*Recall*	*F-Score*
YT001-a	85.60	71.60	17.98	**79.54**	**29.28**
YT001-b	99.90	77.60	21.01	40.90	27.76
YT001-c	**100**	**78.40**	14.64	39.39	21.35
YT001-d	88.06	74.6	27.23	11.36	16.04
YT001-e	99.40	76.00	19.40	54.54	28.62
YT001-f	97.23	77.20	16.75	22.72	19.29
YT001-g	99.80	76.40	**28.09**	25.75	26.87
YT001-h	98.32	74.60	27.41	12.87	17.52
YT001-i	98.71	70.0	22.60	19.69	21.05
YT001-j	95.56	74.6	21.69	17.42	19.32
Média	96.26	75.10	21.68	32.42	22.71
Desvio Padrão	4.92	2.51	4.46	20.44	4.69

Tabela 5.2: Experiência YT001, feito com o *Dataset* V1, com validação, sem *Save-Best*, todos os resultados em %

Nesta primeira análise, tendo em conta apenas uma experiência realizada, é possível verificar a distinção entre os treinos. Apesar de cada uma das execuções utilizar o mesmo *dataset* e o modelo ter sido treinado da mesma maneira, os pesos do modelo foram inicializados de forma aleatória para cada uma das experiências. Isto leva a esta diferença entre os treinos, mesmo que se usem as mesmas condições. Para garantir que, os resultados obtidos são realísticos, para cada experiência são feitas dez execuções, como se pode ver na tabela 5.2. Cada uma das execuções tem como ID o nome da experiência seguido de uma letra.

5.1.2 Avaliação da Experiência YT002

Esta experiência é exactamente igual à anterior, mas com a gravação da melhor versão do modelo, isto é, utilizando o método *Save-Best*. Nesta situação pretende-se perceber se ao utilizar as melhores versões do modelo no teste, isso será traduzido em melhores resultados. Supõe-se que, o modelo final seja melhor que na experiência anterior (YT001), e que, o desvio padrão seja mais pequeno.

Foram realizadas dez novas execuções para este método, como se pode verificar na tabela 5.3. Cada uma das execuções tem como ID o nome da experiência seguido de uma letra, tal como anteriormente definido. É de notar que, novamente os valores são diferentes, sendo a razão a mesma - os pesos do modelo foram inicializados de forma aleatória para cada uma das experiências.

ID	Treino	Validação	Teste (Filme)		
	Precisão		Precisão	*Recall*	*F-Score*
YT002-a	**100**	76.40	15.76	65.90	25.43
YT002-b	94.28	76.00	**29.12**	40.15	**33.75**
YT002-c	99.70	77.20	13.77	50.00	21.60
YT002-d	96.35	76.80	16.98	47.72	25.04
YT002-e	98.81	74.60	21.18	43.18	28.42
YT002-f	86.98	69.80	14.49	43.93	21.80
YT002-g	98.61	75.00	17.14	31.81	22.28
YT002-h	71.10	63.60	11.39	16.66	13.53
YT002-i	**100**	**79.20**	18.70	54.54	27.85
YT002-j	99.40	74.70	13.48	**76.51**	22.92
Média	94.52	74.29	17.20	47.04	24.26
Desvio Padrão	8.68	4.25	4.78	15.86	5.06

Tabela 5.3: Experiência YT002, com o *Dataset* V1, com *Save-Best* e conjunto de validação, todos os resultados em %

Na secção seguinte, será então feita a comparação entre as experiências YT001 e YT002 em maior pormenor.

5.1.3 Comparação entre as experiências YT001 e YT002

Para facilitar a comparação entre as duas experiências, irão ser utilizados os valores que foram obtidos pelo Teste, aplicando média e desvio padrão da distribuição normal. Isso permite então verificar as diferenças entre a Precisão, o *Recall* e o *F-Score*. O método de comparação a qual será dado maior peso é o *F-Score*, sendo aquele que toma em consideração tanto a Precisão do teste, bem como o *Recall*.

É de relembrar que para esta comparação, supôs-se que o YT002 terá melhores resultados, ou seja, médias maiores e desvios padrão mais pequenos que a experiência YT001.

		Precisão		Teste		
		Treino	Validação	Precisão	*Recall*	*F-Score*
YT001	Média	**96.26**	**75.10**	**21.68**	32.42	22.71
	Desvio Padrão	4.92	2.51	4.46	20.44	4.69
YT002	Média	94.52	74.29	17.20	**47.04**	**24.26**
	Desvio Padrão	8.68	4.25	4.78	15.86	5.06

Tabela 5.4: Comparação entre YT001 e YT002, todos os resultados em %

Como é possível ver na tabela 5.4 a previsão sobre o *Save-Best* foi confirmada. O uso da melhor versão do modelo produziu melhores resultados no teste, tendo havido um aumento quase de 2% na média do *F-Score* na experiência YT002. Este aumento deve-se especialmente ao aumento da média *Recall*, sendo que houve um acréscimo de 15%. Isto

quer dizer que o modelo treinado sobre a condição de *Save-Best* é consideravelmente melhor a classificar correctamente a classe positiva durante o teste. Pois, ao apresentar uma média mais alta de *Recall*, indica que está aproximar-se mais dos 100%. Caso atingisse o valor 100%, isto significaria que conseguiu identificar todos os positivos como positivos, independentemente do segmentos de classe negativa que classificou como positivos.

Nestes dois casos, a experiência YT001 obteve apenas 32.42% de *Recall*, enquanto a experiência YT002 conseguiu melhorar o valor, atingindo 47.04%. Por outro lado, quando se trata de Precisão, esta testa a quantidade de previsões que estão realmente correctas, sendo estas sobre os positivos ou negativos. Tal como no *Recall*, quanto mais próximo de 100%, mais correctamente os dados são identificados. Isto significa que o modelo YT001, ao todo, acertou em mais classificações

A segunda previsão não se confirmou. O desvio padrão aumentou em todas as métricas tanto no treino como no teste, excepto no *Recall*.

5.1.4 Avaliação da Experiência YT003

O objectivo desta experiência é verificar a utilidade do processo de validação durante o treino do modelo. Neste caso, não foi feita a validação, tendo sido utilizado um *dataset* de treino com os dados todos (1514 troços). Um segundo objectivo é o estudo do efeito do número de dados presentes num *dataset*, sendo que o esperado é que quando número de dados aumenta, o modelo melhore.

ID	Treino	Validação	Teste (Filme)		
	Precisão		Precisão	*Recall*	*F-Score*
YT003-a	96.16	-	23.45	40.15	29.60
YT003-b	96.56	**-**	18.55	31.06	23.22
YT003-c	93.92	**-**	**36.14**	22.72	27.90
YT003-d	**98.61**	-	24.83	28.78	26.66
YT003-e	98.21	-	13.63	59.09	22.15
YT003-f	79.45	-	20.07	**82.57**	32.29
YT003-g	89.56	-	26.79	65.15	**37.96**
YT003-h	94.45	-	35.29	18.18	24.00
YT003-i	89.29		21.89	66.66	32.95
YT003-j	95.37	-	16.39	61.36	25.97
Média	93.16	-	23.70	47.57	28.26
Desvio Padrão	5.47	-	7.06	20.94	4.73

Tabela 5.5: Experência YT003, feito com o *Dataset* V1, sem validação nem *Save-Best*, todos os resultados em %

Como não foi utilizado um conjunto de validação, como se pode ver na tabela 5.5, não existe resultados de validação.

5.1.5 Comparação entre experiências YT001 e YT003

Esta comparação é feita com a experiência YT001, pois esta não utiliza a condição *Save-Best*, tal como o YT003, apesar de ter obtido piores resultados do que a experiência YT002. Posto que no YT003 não existe conjunto de validação, não é possível comparar os resultados da validação.

		Precisão		Teste		
		Treino	Validação	Precisão	*Recall*	*F-Score*
YT001	Média	**96.26**	75.1	21.69	32.42	22.72
	Desvio Padrão	4.92	2.51	4.47	20.45	4.69
YT003	Média	93.16	-	**23.70**	**47.57**	**28.26**
	Desvio Padrão	5.47	-	7.06	20.94	4.73

Tabela 5.6: Comparação entre YT001 e YT003, todos os resultados em %

A tabela 5.6 ilustra bem o ganho que se obtém ao utilizar mais dados no treino. Em todas as métricas de teste (Precisão, *Recall*, e *F-Score*), a experiência YT003 foi melhor que a experiência YT001. Tal como esperado, durante o treino a precisão dos modelos obtidos pela experiência YT003 obtiveram piores resultados que a experiência YT001, devido a não ser feito a validação dos parâmetros do modelo. Em torno, o desvio padrão da experiência YT003 durante o teste aumentou em todas as métricas, tendo crescido substancialmente na Precisão (cerca de 3%). Apesar disto, os modelos obtidos pelo YT003 são bastante melhores a identificar a correctamente a classe positiva durante o teste, sendo que o *Recall* melhorado cerca de 15%.

Ao todo, é possível dizer que os modelos treinados com um *dataset* sem validação mostram que, ao acrescentar mais dados ao *dataset* os resultados no teste melhoram, e ao mesmo tempo, o facto de não se validar as afinações aos parâmetros do modelo pode causar diferenças grandes entre os resultados do teste. Devido a este último facto, foi escolhido para as restantes experiências utilizar um conjunto de validação.

5.1.6 Variação no *Dataset* V1

Um dos motivos das diferenças entre as experiências, para além das diferenças na composição do *dataset*, pode ser a divisão dos troços entre treino e validação. Para verificar este ponto foram produzidos quatro *datasets* extra, com base no *Dataset* V1, em que a única alteração foi a divisão dos troços entre os dois *datasets* (treino e validação). Assim, a atribuição de um segmento a um dos *datasets* é aleatória.

A partir desta experiência foram utilizadas sempre cinco variações do mesmo *dataset*. Este processo serve também para confirmar os resultados. Nestes casos, apenas apresentamos a média de cada experiência, bem como a média de todas as experiências e o desvio padrão de todas as experiências.

Para esta variação foi utilizado o mesmo *setup* do YT001, ou seja, não é salvo a melhor versão do modelo (*Save-Best*). Como já tinha sido feito o treino da experiência YT001, foram produzidas mais quatro experiências (YT004, YT005, YT006 e YT007) com diferentes divisões nos *datasets* de treino e validação.

ID	Precisão Treino	Precisão Validação	Precisão Teste	*Recall*	*F-Score*
YT001	96.26	**75.10**	21.69	32.42	22.72
YT004	95.38	74.62	18.01	33.11	21.76
YT005	**96.77**	74.70	**21.92**	32.58	24.69
YT006	96.33	73.22	21.51	49.24	**27.46**
YT007	92.54	72.20	16.05	**54.17**	20.75
Média	95.45	73.96	19.83	40.30	23.47
Desvio Padrão	1.70	1.21	2.65	10.55	2.65

Tabela 5.7: Resultados das execuções feitas com o *Dataset* V1, sem *Save-Best*, e com conjunto de validação, todos os resultados em %

Foi realizado mais do que uma experiência para cada *dataset*, devido à maneira de inicializar os pesos do modelo, cada treino do modelo será diferente. Assim, produzimos um conjunto de experiências de modo a analisar se o resultado é o esperado.

Na tabela 5.7 é possível observar a variação entre as diferentes repartições do *dataset* nos conjuntos de treino e validação. Neste caso, a métrica que teve maior variação foi o *Recall*, com uma diferença de 10% entre as diferentes repartições. Outro ponto é a diminuição do desvio padrão.

Na tabela 5.2 é possível ver que o desvio padrão do *Recall* na experiência YT001 é 20.44%, enquanto ao repetir a experiência mais quatro vezes, em que cada experiência têm dez execuções (para um total de 50 execuções), o desvio padrão do *Recall* diminuí para 10.55%. Esta variação entre as experiências também se reflete nas médias. Apesar da média da precisão de teste ter diminuído de 21.69% no YT001 para 19.83% no conjunto das experiências, o *Recall* e em especial o *F-Score* aumentaram.

5.1.7 Análise de Falsos Positivos

Para ser possível produzir um *dataset* que reflita um melhor resultado do modelo, é necessário analisar os Falsos Positivos. Para isso, é calculado a percentagem de segmentos do filme com classe negativa que foram classificados como positivos. Para além disso, é feito o desvio padrão das médias de falsos positivos das diferentes experiências. Para esta análise, foram utilizadas as experiências YT001, YT004, YT005, YT006 e YT007, às quais referimos como os resultados do *Dataset* V1.

Como foi explicado anteriormente, a cada um dos segmentos do filme foi atribuída uma etiqueta, que existe no AudioSet. Ao fazer o teste do modelo foi guardada uma lista dos segmentos que deveriam ser negativos e que o modelo classificou como positivos.

Assim, é feita a tradução para uma das etiquetas apresentadas em cima. Quando um segmento não se inseria em nenhuma etiqueta foi classificado como *Other*.

Dataset V1		
	Média (%)	Desvio Padrão (%)
Screaming	0.00	0.00
Yell	34.62	19.64
Gunshot	20.71	13.34
Music	10.75	14.18
Speech	21.97	18.57
Singing	15.18	12.83
Camera	30.94	21.35
Run	0.00	0.00
Silence	13.65	24.57
Laughter	77.14	41.99
Other	24.62	19.48

Tabela 5.8: Falsos positivos classificados pelos modelos treinados com o *Dataset* V1, sem *Save-Best*

Assim, a tabela 5.8 demonstra a percentagem de segmentos de uma dada etiqueta que foi identificado como positivo, pelos modelos treinados com o *Dataset* V1. Esta percentagem é uma média de todas as experiências. A coluna do desvio padrão indica o desvio padrão do número de segmentos de uma certa etiqueta que foi classificado como FP, entre todas as experiências. Existem três etiquetas que se encontram bastante superiores às restantes: *Laughter*, *Camera*, *Yell*.

É de notar que existe apenas um segmento do filme com *Laughter*, é normal que a média seja tão alta. Neste caso, isto quer dizer que das 50 execuções que houve (5 experiêcias e 10 execuções por experiência), 77.14% classificou o segmento que deveria ser *Laughter* como Positivo. Também, é normal que a etiqueta *Run* seja sempre 0 pois não há um segmento com *Run*.

5.2 *Dataset* V2

Após analisar os Falsos Positivos dos modelos criados com o *Dataset* V1, um dos Falsos Positivos que mais vezes ocorre é o som de máquinas fotográficas. Este foi o escolhido por ser um som mais identificável e distinto dos outros dois que foram indicados como problemáticos (*Laughter* e *Yell*). Na sequência disto, foi criado um *dataset* com base no *dataset* V1 ao qual foram adicionados troços com sons de máquinas fotográficas. Estas classes estão presentes no AudioSet como *Camera* e *Single-Lens Camera*.

Nome	Classes Positivas		Classes Negativas		Total
	Classes	# de Troços	Classes	# de Troços	
V2	*Screaming*	758	*Speech*, *Singing*, *Yell*, *Run*, *Camera*, *Single-Lens*	756 (126 por classe)	1514

Tabela 5.9: Constituição do *Dataset* V2

Com estas especificações foram criadas cinco variações do mesmo *dataset*, com diferentes repartições entre o conjunto de treino e o de validação.

5.2.1 Avaliação da Experiência YT008

Nesta experiência pretende-se compreender se o modelo melhora a sua classificação de sons de *Camera* no teste, e se ao aumentar o número de dados os resultados do teste também melhoram. Para este teste não foi utilizado a condição *Save-Best*, isto é, foi guardado o modelo da última iteração de treino, e não o melhor.

Sendo que o número de dados presente no *dataset* é igual ao *Dataset* V1, o esperado é que o modelo treinado com este *dataset* tenha obtido valores semelhantes aos modelos treinados com o *Dataset* V1, com a excepção do *Recall*, pois foi acrescentado mais *labels* à classe negativa. Ao comparar a tabela 5.10 com a tabela 5.2 é possível ver um acréscimo substancial em todas as métricas de teste. A média da Precisão de teste aumentou 10%, enquanto o *Recall* aumentou cerca de 6%. Sendo que o *F-Score*, sendo uma medida que toma em consideração as duas métricas anteriores, teve por um acréscimo de cerca 5%. O desvio padrão aumentou também em todas as métricas de teste, tendo tido um acréscimo mais significativo na Precisão, onde passou de 4.46% para 11.66%.

5.2.2 Variação da Experiência YT008

Apesar dos resultados obtidos na experiência anterior (YT008) não serem muito maus em média, o desvio padrão aumentou. De modo a retirar valores mais baixos dos parâmetros estatísticos, foram feitas mais quatro experiências.

	Treino	Validação	Teste (Filme)		
ID	Precisão		Precisão	*Recall*	*F-Score*
YT008-a	98.91	76.00	23.54	55.30	33.03
YT008-b	97.63	75.2	17.00	57.57	26.25
YT008-c	90.72	76.80	35.25	9.09	14.45
YT008-d	88.85	66.40	12.67	**88.63**	22.18
YT008-e	97.53	74.60	**46.66**	15.90	23.72
YT008-f	99.40	77.00	35.59	31.81	33.6
YT008-g	92.40	**78.4**	44.23	17.42	25.00
YT008-h	**100**	77.40	31.03	47.72	**37.61**
YT008-i	99.50	78.60	25.11	42.42	31.54
YT008-j	95.95	76.80	47.91	17.42	25.55
Média	96.09	75.72	31.91	38.33	27.30
Desvio Padrão	3.81	3.33	11.66	23.6	6.41

Tabela 5.10: Resultados obtidos no treino da experiência YT008, todos os resultados em %

ID	Precisão Treino	Precisão Validação	Precisão Teste	*Recall*	*F-Score*
YT008	96.09	75.72	31.91	38.33	27.3
YT009	**98.54**	**76.88**	31.44	32.5	28.88
YT010	95.37	73.62	29.91	38.86	**30.20**
YT011	95.57	74.82	**32.57**	33.71	27.24
YT012	95.92	75.98	28.89	**39.32**	26.48
Média	96.29	75.40	30.94	36.54	28.02
Desvio Padrão	1.28	1.23	1.50	3.18	1.49

Tabela 5.11: Variação das experiências feitas com base no YT008, todos os resultados em %

Novamente, para cada experiência, foi feita uma média e calculado o desvio padrão, e é isso que a tabela 5.11 representa. Com a mesma, pode-se concluir que o *Dataset* V2 apresenta uma melhoria do modelos treinados. Estes modelos, quando comparados com as experiências do *Dataset* V1, melhoraram em todas as métricas, excepto no *Recall*.

		Precisão		Teste		
		Treino	Validação	Precisão	*Recall*	*F-Score*
V1	Média	95.46	73.97	19.84	**40.30**	23.48
	Desvio Padrão	7.81	4.57	4.00	28.72	6.93
V2	Média	**96.29**	**75.40**	**30.94**	36.54	**28.02**
	Desvio Padrão	1.28	1.23	1.50	3.18	1.49

Tabela 5.12: Comparação entre *Dataset* V1 e *Dataset* V2, sem *Save-Best*, todos os resultados em %

Comparando então o *Dataset* V1 e V2, utilizando a tabela 5.12, é possível ver que o desvio padrão diminuiu consideravelmente em todas as métricas, tanto no treino como no teste. Primeiro, os modelos treinados com o *Dataset* V2 conseguiram melhores resultados de treino, tendo melhorado a precisão de treino de 95.46% para 96.29%. No entanto, mais importante que isso, é a diminuição drástica do desvio padrão, passando este de 7.81% para 1.28%. O mesmo verifica-se nos resultados da precisão na validação, pois a média aumentou e o desvio padrão diminuiu.

Os resultados tornam-se realmente interessantes no teste. Como foi referido, para a construção do *Dataset* V2 foram adicionados duas *labels*, mas foi mantido o mesmo número de dados no *dataset*. Apesar disto, o *Recall* das experiências do *Dataset* V2 foi menor que as do *Dataset* V1. Ao mesmo tempo, houve uma diminuição significativa do desvio padrão, passando de 28.72% para 3.18%, que representa uma redução de 25%. Ainda assim, a precisão do teste aumentou consideravelmente, passando de 19.84% para 36.54%, o que em torno, fez com que o *F-Score* tenha também aumentado.

No que toca ao desvio padrão, este já é bastante reduzido e, acredita-se que mesmo que aumentasse o número de experiências, esse valor não iria diminuir muito mais.

5.2.3 Análise de Falsos Positivos

Como foram acrescentadas mais duas etiquetas à classe negativa, é necessário analisar os falsos positivos produzidos pelos modelos das experiências treinados com o *Dataset* V2. De novo, será comparado a média da quantidade de falsos positivos para as *labels* da classe negativa para cada uma das experiências, e o desvio padrão das mesmas.

Como é possível ver na tabela 5.13, em média, os modelos treinados com o *Dataset* V2 classificam melhor os segmentos com etiqueta *Camera*, tendo diminuído de 30.93% para 15.49%. Isto deve-se a introdução de dados que não se encontravam no *Dataset* V1, que levou um aumento no *Recall* que passou de 19% para 30%. Ao comparar com os

	Dataset V1	Dataset V2	
	Média (%)	Média (%)	Desvio Padrão (%)
Screaming	0.00	0.00	0.00
Yell	34.61	24.53	21.20
Gunshot	20.71	12.89	11.02
Music	10.75	9.36	16.45
Speech	21.97	11.73	16.96
Singing	15.17	9.50	12.13
Camera	30.93	14.69	16.52
Run	0.00	0.00	0.00
Silence	13.65	4.22	14.04
Laughter	77.14	38.00	48.54
Other	24.62	15.95	18.63

Tabela 5.13: Falsos Positivos do *Dataset* V2 em comparação com o *Dataset* V1

resultados obtidos do *Dataset* V1, é possível ver que as foi obtido melhores resultados em todas as *labels*, claramente devido à introdução de mais etiquetas, que permitiram uma melhor identificação dos segmentos levando a diminuir os falsos positivos.

Apesar dos resultados terem melhorado, não é viável aumentar demasiado o número de *labels* na classe negativa, se mantiver o número de troços na classe positiva. Pois, isso pode levar à diminuição de troços por etiqueta na classe negativa, dificultando o processo de treino. Contudo, neste contexto, o aumento de *labels* foi suficiente para verificar melhoria nos resultados.

5.2.4 Avaliação da Experiência YT013

De modo a confirmar que, ao utilizar o *Save-Best* são obtidos melhores resultados, a experiência anterior (YT008) foi repetida, mas com a condição *Save-Best*.

Olhando individualmente para a Precisão e *Recall* nas duas experiências (YT008 e YT013), é possível verificar que ambos diminuíram para a experiência *Save-Best*. No entanto, devido ao método de cálculo do *F-Score*, que toma em consideração tanto a Precisão como o *Recall*, estes, ao apresentar números mais próximos, acabam por apresentar o valor do *F-Score* maior. Na experiência YT008, a diferença entre os valores médios era de 7% ($31.91\% - 38.77\%$) resultando num *F-Score* de 27.30%. Enquanto neste caso foi de apenas de 2% ($30.58 - 32.88\%$), resultando num *F-Score* de 30%, como é possível ver na tabela 5.14. No que toca o desvio padrão, com o método *Save-Best* a percentagem do mesmo diminuiu para as três métricas do teste.

	Treino	Validação	Teste (Filme)		
ID	Precisão		Precisão	*Recall*	*F-Score*
YT013-a	93.49	77.6	26.54	42.42	32.65
YT013-b	89.25	65.8	32.17	34.85	**33.45**
YT013-c	87.48	74.8	**42.17**	26.52	32.56
YT013-d	97.44	77.6	33.61	30.30 0	31.87
YT013-e	98.42	77.6	22.01	**53.03**	31.11
YT013-f	95.07	70.6	17.96	22.73	20.07
YT013-g	95.56	76.8	38.89	26.52	31.53
YT013-h	96.15	75.4	29.63	36.36	32.65
YT013-i	97.34	78.6	29.88	37.12	33.11
YT013-j	**99.41**	**80.20**	32.89	18.94	24.04
Média	94.96	75.50	30.58	32.88	30.30
Desvio Padrão	3.69	4.06	6.86	9.57	4.27

Tabela 5.14: Resultados das execuções da experiência YT013, sobre regime *Save-Best*, todos os resultados em %

5.2.5 Variação da Experiência YT013

Visto que, na secção 5.2.2, ao serem feitos mais quatro experiências baseadas no YT008, houve uma diminuição drástica do desvio padrão, foi feito o mesmo, para a condição *Save-Best*, isto é, mais quatro execuções baseadas no YT013.

	Precisão Treino	Precisão Validação	Precisão Teste	*Recall*	*F-Score*
YT013	94.96	**75.50**	30.58	32.88	30.30
YT014	93.15	72.24	28.79	32.42	28.55
YT015	**97.16**	74.94	**33.09**	31.89	**31.20**
YT016	95.28	73.80	31.12	33.33	28.92
YT017	95.12	74.62	27.27	**37.95**	30.99
Média	95.13	74.22	30.17	33.70	30.00
Desvio Padrão	6.03	3.25	6.57	14.54	6.05

Tabela 5.15: Variação dos resultados obtidos com a experiência YT013, todos os resultados em %

Na tabela 5.15 , é possível observar que, ao serem feitos mais quatro execuções com base na experiência YT013, os resultados não apresentam variações significativas em comparação com os valores da tabela 5.14. Comparando com a tabela 5.11, onde também foram feitas mais quatro experiências, mas usando a base de YT008, é possível ver que os desvios padrão são superiores nesta situação.

No entanto, toma-se apenas em conta o *F-Score*, e sendo assim, na condição *Save-Best* a média dessa métrica é melhor, logo este método é melhor a identificar o resultado.

5.2.6 Análise de Falsos Positivos

Dado que, ao utilizar o *Save-Best* foram obtidos melhores resultados , ao analisar os falsos positivos deste modelos, é esperado que haja uma diminuição dos mesmos.

	Sem Save-Best	Com Save-Best	
	Média (%)	Média (%)	Desvio Padrão (%)
Screaming	0.00	0.00	0.00
Yell	24.53	22.29	10.29
Gunshot	12.89	11.00	7.07
Music	9.36	4.31	4.17
Speech	11.73	7.88	5.18
Singing	9.50	8.25	7.75
Camera	14.69	10.63	5.52
Run	0.00	0.00	0.00
Silence	4.22	0.44	2.18
Laughter	38.00	44.00	49.64
Other	15.95	11.79	6.03

Tabela 5.16: Percentagem de falsos positivos de cada etiqueta, do conjunto de experiências treinadas com o *Dataset* V2, com *Save-Best*

Tal como esperado, a tabela 5.16 ilustra esta diminuição , em cada um das *labels*. Em especial, é de notar que a diminuição que a etiqueta *Camera* teve foi a mais significativa. Assim, para o *Dataset* V2, o uso de *Save-Best* levou a uma melhoria dos resultados obtidos, ao testar com o filme.

5.3 *Dataset* V3

Na construção do *Dataset* V2 foi mantido o mesmo número de troços por classe como no *Dataset* V1, o que levou à diminuição de troços em cada etiqueta na classe negativa, passando de 189 para 126. Na construção do *Dataset* V3, foi mantido o mesmo número de troços em cada etiqueta da classe negativa do *Dataset* V1 (189). Assim, será avaliado o efeito do treino com *datasets* desequilibrados, isto é, um *dataset* que as duas classes não têm o mesmo número de de troços.

Nome	Classes Positivas		Classes Negativas		Total
	Classes	# de Troços	Classes	# de Troços	
V3	*Screaming*	758	*Speech*, *Singing*, *Yell*, *Run*, *Camera*, *Single-Lens*	1134 (189 por classe)	1892

Tabela 5.17: Constituição do *Dataset* V3

Desta forma, é esperado que as médias nos diferentes critérios de comparação no teste, sejam maiores do que os obtidos pelas experiências do *Dataset* V2, devido ao aumento no número de dados, e que os desvios padrão destes mesmo critérios sejam também superiores aos conseguidos pelo *Dataset* V2. devido à diferença de troços entre as classes. Pode ser esperado que o modelo seja melhor a classificar correctamente a classe positiva, resultando num *Recall* mais elevado. Também será analisado o resultado de aplicar a condição *Save-Best* com *datasets* irregulares. É expectável que o resultado, ao usar a condição *Save-Best*, seja melhor do que não utilizar a condição, devido apenas ser utilizado os melhores modelos para o teste. É também esperado que a variação entre execuções da experiência que utiliza esta condição seja menor.

5.3.1 Avaliação da Experiência YT018

Dado que os efeitos do treino usando um *dataset* desequilibrado ainda não foram estudados, o objectivo desta experiência é mesmo esse. É expectável resultados um pouco imprevisíveis, dado a natureza do *dataset* utilizado. Para esta primeira experiência não foi aplicada a condição *Save-Best*.

Após uma comparação rápida entre os resultados da tabela 5.18 e os resultados da tabela 5.10 é possível ver o efeito do uso de um *dataset* irregular. Os resultados médios obtidos na experiência YT018 em cada uma das métricas são o oposto dos obtidos pela experiência YT008, onde a média da precisão de treino, *Recall* e *F-Score* são superiores aos do YT018. Depois, os desvios padrão em cada uma das métricas da experiência YT008 são menores do que aqueles conseguidos pela experiência YT018.

	Treino	Validação	Teste (Filme)		
ID	Precisão		Precisão	*Recall*	*F-Score*
YT018-a	98.18	79.52	27.90	27.27	27.58
YT018-b	96.21	76.00	19.36	37.12	25.45
YT018-c	90.37	72.16	30.61	**45.45**	**36.58**
YT018-d	98.89	**79.52**	43.58	12.87	19.88
YT018-e	68.35	67.36	**100**	0.75	1.50
YT018-f	96.68	79.36	28.26	9.85	14.60
YT018-g	91.63	73.76	17.79	37.87	24.21
YT018-h	98.50	77.12	25.25	37.12	30.06
YT018-i	**99.13**	78.56	22.76	40.15	29.04
YT018-j	98.18	79.36	22.74	40.15	29.04
Média	93.68	76.27	33.82	28.86	23.79
Desvio Padrão	8.93	3.86	23.1	14.69	9.34

Tabela 5.18: Experiência YT018 - onde o *dataset* é irregular e escolhe sempre o último resultado, todos os valores em %

Neste caso, ter sido acrescentado mais troços ao *dataset* não sucedeu-se de ter melhores médias no teste. Isto deve-se à natureza irregular do *dataset*.

5.3.2 Variação do *Dataset* V3

Numa tentativa de estudar o efeito da variação dos resultados, obtidos por diferentes experiências com um *dataset* irregular, foram feitas mais quatro experiências. Tal como no YT018, cada uma destas experiências não ocorreu sobre regime *Save-Best*.

ID	Precisão Treino	Precisão Validação	Precisão Teste	*Recall*	*F-Score*
YT018	93.69	76.27	33.83	28.86	23.80
YT019	**96.88**	**77.17**	**34.56**	22.42	24.31
YT020	94.05	75.14	30.68	**42.20**	**30.78**
YT021	95.11	75.06	31.16	37.27	29.38
YT022	96.50	75.95	29.24	34.17	28.99
Média	95.24	75.91	31.89	32.98	27.45
Desvio Padrão	5.69	3.17	8.00	22.35	7.94

Tabela 5.19: Todas as experiências realizadas utilizando o *dataset* V3 (guarda último resultado), todos os valores em %

Apesar de, no conjunto das 5 experiências, cada experiência ter conseguido desvios padrão menores em algumas métricas, ao comparar a tabela 5.19 com a tabela 5.11 é claro que, o uso de *datasets* irregulares não é indicado, pois o aumentar de dados não resultou em médias melhores, tendo o *F-Score* diminuído de 28.02% para 27.45%, e em especial, o *Recall* diminuído de 36.54% para 32.98%. Esta última métrica é importante, pois neste

dataset havia mais dados na classe negativa, e claramente os modelos não são melhores a classificar os segmentos do filme que deveriam ser negativos.

5.3.3 Análise de Falsos Positivos

Visto que, o *dataset* utilizado para este conjunto de experiências não é regular, e os valores conseguidos pelos modelos treinados por este *dataset* foram piores do que os conseguidos pelo conjunto de experiências do *Dataset* V2, o esperado é que em média, tenha sido classificado mais falsos positivos para todas as *labels*.

	Dataset V2	Dataset V3	
	Média (%)	Média (%)	Desvio Padrão (%)
Screaming	0.00	0.00	0.00
Yell	24.53	21.05	16.40
Gunshot	12.89	10.44	9.14
Music	9.36	6.39	11.69
Speech	11.73	8.52	9.69
Singing	9.50	8.00	10.83
Camera	14.69	12.06	10.63
Run	0.00	0.00	0.00
Silence	4.22	3.56	7.84
Laughter	38.00	42.00	49.36
Other	15.95	12.87	12.65

Tabela 5.20: Falsos Positivos do Dataset V3

Os falsos positivos diminuíram em todas as *labels*, algo que é contraditório numa primeira análise, sendo que, as experiências feitas com o *Dataset* V3 tiveram piores resultados no teste do que as experiências feitas com o *Dataset* V2. Mas, sendo que a precisão média do teste melhorou das experiências do *Dataset* V2 para o V3, o modelo tornou-se melhor a identificar os positivos, o que levou também uma diminuição de falsos positivos.

5.3.4 Avaliação da Experiência YT023

Sendo que, o desvio padrão, no conjunto de experiências feitas sobre o *Dataset* V3 sem a condição de *Save-Best*, é elevado, um passo lógico é a execução do mesmo conjunto de experiências, mas sobre a condição de *Save-Best*.

Ao comparar a tabela 5.14 com a tabela 5.21 é possível ver o efeito que o uso da condição *Save-Best* pode ter, ao ser utilizado no treino com um *dataset* irregular. Apesar de não ter sido obtido melhores resultados, os desvios padrão sobre os resultados do teste diminuíram.

	Treino	Validação	Teste (Filme)		
ID	Precisão		Precisão	*Recall*	*F-Score*
YT023-a	96.60	79.36	**39.47**	22.72	**28.84**
YT023-b	89.81	77.75	26.19	16.66	20.37
YT023-c	83.74	73.92	34.40	24.24	28.44
YT023-d	85.00	66.24	18.12	20.45	19.21
YT023-e	85.16	76.32	23.94	25.75	24.81
YT023-f	90.84	78.24	18.79	40.15	25.60
YT023-g	96.68	76.16	30.90	12.87	18.18
YT023-h	97.86	80.47	23.30	21.96	22.56
YT023-i	98.26	79.67	30.61	11.36	16.57
YT023-j	**99.21**	**80.47**	17.84	**51,51**	26.51
Média	92.32	76.86	26.35	24.77	23.11
Desvio Padrão	5.81	4.07	6.99	11.71	4.15

Tabela 5.21: Experiência YT023 onde dataset é irregular e utiliza o método Save Best, todos os valores em %

5.3.5 Variação do *Dataset* V3 com *Save-Best*

Novamente, foram feitas mais cinco experiências sobre o *Dataset* V3, na condição de *Save-Best*. Estas experiências vêm clarificar os pontos observados anteriormente, de que um *dataset* irregular não produz resultados satisfatórios ao ser testado com um filme.

ID	Precisão Treino	Precisão Validação	Precisão Teste	*Recall*	*F-Score*
YT023	92.32	76.86	26.35	**24.77**	23.11
YT024	94.59	74.80	26.35	**24.77**	23.11
YT025	97.71	**77.33**	**33.39**	**24.77**	**27.52**
YT026	96.32	75.34	32.06	22.72	25.18
YT027	**98.02**	77.07	31.27	24.54	27.01
Média	95.79	76.28	30.81	23.17	25.11
Desvio Padrão	1.73	1.73	6.87	7.22	5.52

Tabela 5.22: Todas as experiências realizadas utilizando *dataset* V3 (*Save Best*), todos os valores em %

Ao analisar a tabela 5.22 é possível ver que os resultados, ao comparar com os obtidos sem o uso da condição *Save-Best*, presentes na tabela 5.19, são piores, apesar de terem desvios padrão mais pequenos.

A tabela 5.23 confirma o ponto anteriormente descrito. Os resultados obtidos ao utilizar a condição *Save-Best* são piores em todas as métrica de teste. Em especial, houve um decréscimo significativo no *Recall*, que desceu de 31.89% para 23.17%. O decréscimo da Precisão e *F-Score* foi menor, mas mesmo assim significativo.

Apesar de ao não usar a condição *Save-Best* foram conseguidos melhores resultados (*F-Score* superior no caso de *Save-Best false*), estes variam mais, tendo um desvio padrão

		Precisão Treino	Precisão Validação	Precisão Teste	*Recall*	*F-Score*
Save-Best False	Média	95.24	75.91	31.89	32.98	27.45
	Desvio Padrão	5.81	4.07	6.99	11.71	4.15
Save-Best True	Média	95.79	76.28	30.81	23.17	25.11
	Desvio Padrão	1.73	1.73	6.87	7.22	5.52

Tabela 5.23: Comparação da opção *Save-Best*, com o *Dataset* V3

maior, particularmente no *Recall*, onde o desvio padrão foi de 11.71%, enquanto ao usar a condição *Save-Best* foi conseguido um desvio padrão de 7.22%. Além disto, os desvios padrão do treino também diminuíram consideravelmente.

Para concluir, a utilização de *datasets* irregulares não é recomendada e percebe-se porquê. Até este ponto, todas as experiências feitas tiveram resultados semelhantes: ao aumentar o número de dados e ao usar a condição *Save-Best* os resultados no teste melhoram. Os dois conjuntos de experiências revelaram que com um *dataset* irregular isso não acontece, portanto os *datasets* criados daqui em diante serão sempre regulares.

5.4 *Dataset* V4

Sabendo que os *datasets* irregulares produzem resultados imprevisíveis, foi criado o *Dataset* V4, com base no *Dataset* V3 ao qual foram adicionados mais troços da classe positiva, de modo a igualar as duas classes.

Nome	Classes Positivas		Classes Negativas		Total
	Classes	# de Troços	Classes	# de Troços	
V4	*Screaming*	1134	*Speech*, *Singing*, *Yell*, *Run*, *Camera*, *Single-Lens*	1134 (189 por classe)	2268

Tabela 5.24: Divisão das classes positivas e negativas para o *Dataset* V4

Como já tinham sido utilizados todos os vídeos disponibilizados pelo AudioSet, foram descarregados vídeos extra do Youtube. Estes vídeos são composições de efeitos sonoros, que são utilizados em filmes. Foram utilizados quatro vídeos, e após terem sido descarregados, foi procedido o mesmo tratamento do filme: corte de troços de 10 segundos a cada 5 segundos. Assim, os troços criados estão sobrepostos. Foram adicionados tantos troços quanto os necessários para igualar as duas classes.

No *Dataset* V2 foi concluído que, ao adicionar troços de *labels* que tenham muitos falsos positivos, o modelo melhora a sua classificação dos mesmos. No *Dataset* V3 foi apontado que o uso de *datasets* irregulares causa resultados imprevisíveis. Assim, com este *dataset*, será analisado se, ao adicionar mais dados que não pertencem ao AudioSet e que estão sobrepostos, o modelo tem melhores resultados no teste.

5.4.1 Avaliação da Experiência YT028

Sendo que, os dados introduzidos neste *dataset* não fazem parte do AudioSet, serão feitas duas experiências, uma sobre a condição *Save-Best* e outra sem esta condição. Esta primeira experiência (YT028) não ocorre sobre regime *Save-Best*.

É expectável que ao introduzir mais dados, os resultados melhorem, em especial a precisão do teste, visto que os novos dados fazem parte da classe positiva. Numa outra visão, os números de treino e teste podem piorar, devido aos tipo de dados introduzidos, relembrando que os dados não pertencem ao AudioSet, e têm sobreposição.

Esta experiência será comparada com a experiência YT013, sendo que não foi utilizado a condição *Save-Best*. Numa primeira análise, ao comparar a tabela 5.25 com a tabela 5.18, é notório o efeito dos novos dados. Primeiro, referir que o *F-Score* aumentou, passando de 23.79% para 25.36%. Dado que a precisão de teste piorou (de 33.82% para 27.48%) este aumento do *F-Score* é devido ao aumento drástico do *Recall*, que passou de

	Treino	Validação	Teste (Filme)		
ID	Precisão		Precisão	Recall	F-Score
YT028-a	95.12	83.97	38.09	12.12	18.39
YT028-b	94.73	80.10	27.77	3.78	6.66
YT028-c	**99.73**	82.10	29.24	23.48	26.05
YT028-d	94.40	79.17	23.16	45.45	30.69
YT028-e	89.20	77.70	23,95	60.60	**34.33**
YT028-f	98.88	**85.04**	**40.98**	18.93	25.90
YT028-g	99.47	84.51	29.87	34.84	32.16
YT028-h	95.52	75.96	14.85	67.42	24.35
YT028-i	99.53	83.97	33.33	30.30	31.74
YT028-j	68.66	64.08	13.47	**93.18**	23.54
Média	93.53	79.67	27.48	39.02	25.38
Desvio Padrão	8.86	5.97	8.51	26.36	7.74

Tabela 5.25: Resultados das execuções da experiência YT028, todos os valores em %

28.86% na experiência YT013 para 39.02%.É de referir também que o desvio padrão no *Recall* passou de 14.69% para 26.36%.

5.4.2 Variação do *Dataset* V4 sem *Save-Best*

Para confirmar o efeito que os novos dados tiveram sobre o modelo, foram feitas mais quatro experiências, todas sem a condição *Save-Best*. É esperado que em todas as medidas o desvio padrão diminua consideravelmente.

ID	Precisão Treino	Precisão Validação	Precisão Teste	*Recall*	*F-Score*
YT028	95.05	80.23	30.63	21.97	23.38
YT029	95.77	80.61	26.97	**32.35**	26.98
YT030	94.86	80.96	**38.26**	23.11	28.15
YT031	96.46	80.11	31.32	26.29	26.22
YT032	**98.31**	**82.27**	27.77	32.05	**28.55**
Média	94.12	79.64	29.77	35.55	24.84
Desvio Padrão	8.86	5.97	8.51	26.36	7.74

Tabela 5.26: Resultados das experiências com base na YT028, sem *Save-Best*, todos os valores em %

Ao comparar a tabela 5.26 com a tabela 5.19, apesar deste conjunto de experiências ter conseguido menores desvios padrão em todas as métricas, os resultados do teste foram piores. É de notar o *Recall*, que foi a métrica de teve a maior queda, passando de 32.98% para 27.15%. Por sua vez, o *F-Score* passou de 27.45% no conjunto de experiências do *Dataset* V3, para 24.84% no conjunto de experiências do *Dataset* V4. Isto vai contra o expectado, isto é, quantos mais dados tem o *dataset* melhor é o teste com o filme.

Para além disso, se comparamos este conjunto de experiências com a tabela 5.11, já que ambos *datasets* são semelhantes na sua constituição, é possível observar que os modelos treinados com o *Dataset* V4, apesar de ter mais 1000 troços que o *Dataset* V2, conseguiram piores resultados de teste e um desvio padrão melhor, tendo o *F-Score* diminuído de 28.02% no *Dataset* V2 para 26.66%, e o desvio padrão na mesma métrica ter aumentado de 1.49% para 4.99%.

5.4.3 Análise de Falsos Positivos

Para perceber o porquê dos modelos treinados com o *Dataset* V4 são piores do que os treinados pelo *Dataset* V3, e porque vão contra o suposto que quanto mais dados melhor é resultado, foram analisados os falsos positivos produzidos pelo conjunto de experiências feitas.

	Dataset V3	Dataset V4	
	Média (%)	Média (%)	Desvio Padrão (%)
Screaming	0.00	0.00	0.00
Yell	21.05	24.70	22.59
Gunshot	10.44	12.31	12.46
Music	6.39	9.97	15.40
Speech	8.52	13.17	16.58
Singing	8.00	10.54	12.71
Camera	12.06	16.36	16.10
Run	0.00	0.00	0.00
Silence	3.56	11.76	21.94
Laughter	42.00	47.06	49.91
Other	12.87	17.81	19.41

Tabela 5.27: Falsos Positivos do Dataset V4 sem *Save-Best*

Os resultados da tabela 5.27 eram esperados, dado que a precisão de teste no conjunto de experiências do *Dataset* V4, diminuiu em relação ao conjunto de experiências do *Dataset* V3. Deste modo, os modelos treinados com o *Dataset* V4 são piores a classificar os segmentos do filme. Sendo que, a diminuição foi cerca de 1%, a diferença não é grande e explica as pequenas diferenças em cada uma das *labels.*

5.4.4 Avaliação da Experiência YT028

Após os resultados do *Dataset* V4 sem a condição *Save-Best*, de modo a retirar conclusões sobre o uso de dados extra, que não pertencem ao AudioSet, foi feito um novo conjunto de experiências, desta vez com a condição *Save-Best*.

	Treino	Validação	Teste (Filme)		
ID	Precisão		Precisão	*Recall*	*F-Score*
YT033-a	73.46	68.09	14.81	27.27	19.19
YT033-b	98.42	83.57	30.00	13.63	18.75
YT033-c	**99.80**	82.64	34.69	12.87	18.78
YT033-d	99.60	81.84	30.76	**33.33**	32.00
YT033-e	97.16	81.57	23.54	42.42	**32.65**
YT033-f	95.45	80.24	**50.00**	11.36	18.51
YT033-g	97.82	79.70	27.27	29.45	23.37
YT033-h	98.02	**84.24**	25.0	21.96	23.38
YT033-i	91.24	80.24	39.39	19.69	26.26
YT033-j	99.47	80.10	27.84	16.66	20.83
Média	95.05	80.23	30.63	21.97	23.38
Desvio Padrão	7.59	4.3	8.87	9.36	5.09

Tabela 5.28: Resultados da experiência YT028, todos os valores em %

Tal como na experiência YT002, o usar o *Save-Best* sobre um *dataset* equilibrado, é expectável que os resultados melhorem em relação ao conjunto de experiências anteriores.

Numa primeira análise, ao comparar a tabela 5.28 com a tabela 5.25 não se verifica o aumento no *F-Score* como esperado. Aliás, houve uma descida considerável no resultado do *Recall*, passando de 39.02% na experiência YT023, para 21.97% nesta experiência. Isto quer dizer que este modelo, que foi treinado sobre regime *Save-Best*, é pior a classificar correctamente os segmentos positivos que o modelo da experiência YT023. Além disto, este modelo tem uma precisão de teste maior que o anterior, tendo passado de 27.48% na experiência YT023 para 30.48% no YT028. Provavelmente, tendo a precisão correcta dos segmentos positivos diminuído, e a precisão ao todo aumentado, isto poderá querer dizer que o modelo treinado sobre regime *Save-Best* identificou menos falsos positivos, e por consequente, mais falsos negativos.

Para confirmar o dito aqui, foram feitas mais 4 experiências.

5.4.5 Variação do *Dataset* V4 com *Save-Best*

Dado os resultados que vão contra o esperado, na medida que ao adicionar mais dados o modelo não melhorou os seus resultados, foram feitas mais quatro experiências, para além da experiência YT028, todas sobre o regime *Save-Best*, com o intuito de confirmar os resultados obtidos por esta experiência.

Desde logo, é possível observar, numa análise entre as tabelas 5.29 e a tabela 5.26, que os modelos resultantes destas experiências obtiveram resultados praticamente iguais, tanto no treino como no teste.

Ao comparar a tabela 5.29 com a tabela 5.22 é observado que, ao introduzir dados fora do AudioSet, e que foram sobrepostos, é possível obter melhores resultados que usar um

ID	Precisão Treino	Precisão Validação	Precisão Teste	*Recall*	*F-Score*
YT033	95.05	80.23	30.63	21.97	23.38
YT034	95.77	80.61	26.97	**32.35**	26.98
YT035	94.86	80.96	**38.26**	23.11	28.15
YT036	96.46	80.11	31.32	26.29	26.22
YT037	**98.31**	**82.27**	27.77	32.05	**28.55**
Média	96.09	80.83	30.98	27.55	26.65
Desvio Padrão	1.39	0.86	4.46	5.42	2.04

Tabela 5.29: Resultados do conjunto de experiências feitas com o *Dataset* V4, com Save-Best, todos os valores em %

dataset desequilibrado, se este for treinado sobre regime *Save-Best*, mas esta diferença não é grande. O *F-Score* passou de 25.11% no *Dataset* V3 para 26.65% no *Dataset* V4. É de relembrar que, as experiências do *Dataset* V3 com a condição *Save-Best* tiveram piores resultados do que as experiências do mesmo *dataset* sem a condição, indo contra o esperado. Neste caso, não houve diferença significativa.

		Precisão Treino	Precisão Validação	Precisão Teste	*Recall*	*F-Score*
Save-Best False	Média	94.12	79.64	29.77	35.55	24.84
	Desvio Padrão	8.86	5.97	8.51	26.36	7.74
Save-Best True	Média	96.09	80.83	30.98	27.55	26.65
	Desvio Padrão	1.39	0.86	4.46	5.42	2.04

Tabela 5.30: Comparação da condição *Save-Best* com o Dataset V4, todos os valores em %

A tabela 5.30 ilustra de uma forma sintáctica as duas condições. É possível observar que, apesar de não haver diferenças significativas entre os conjuntos de experiências treinadas com o *Dataset* V4, o desvio padrão diminuiu no conjunto que foi treinado sobre a condição *Save-Best*, algo que é esperado. Ao comparar os dois conjuntos, é também possível ver que o desvio padrão também diminuiu, especialmente nas métricas de teste, onde a precisão baixou de 7.04% no *Dataset* V3 para 4.46% no *Dataset* V4. O *Recall* diminuiu de 9.75% para 5.42%, e o *F-Score* baixou de 4.99% para 2.04%.

5.4.6 Análise de Falsos Positivos

Dado que não existem grandes diferenças entre as experiências treinadas com o *Dataset* V3 e o *Dataset* V4, não é esperado haver grandes diferenças no número médio de falsos positivos nestes dois conjuntos de experiências. A tabela 5.31 ilustra esse ponto. Apesar dos resultados obtidos pelas experiências treinadas usando o *Dataset* V3 são melhores que os obtidos pelas experiências treinadas com o *Dataset* V4, esta diferença não é grande.

A tabela 5.32 ilustra melhor as diferenças entre os dois conjuntos de experiências, sendo que, tal como esperado, o conjunto de experiências treinado sobre *Save-Best* classificou menos falsos positivos.

	Dataset V3	Dataset V4	
	Média (%)	Média (%)	Desvio Padrão (%)
Screaming	0.00	0.00	0.00
Yell	14.55	16.62	8.96
Gunshot	6.22	6.89	7.82
Music	2.97	3.11	5.22
Speech	4.90	6.39	5.64
Singing	3.25	7.50	9.01
Camera	9.25	10.63	6.12
Run	0.00	0.00	0.00
Silence	2.00	4.44	14.57
Laughter	50.00	50.00	50.00
Other	8.06	9.93	6.71

Tabela 5.31: Falsos Positivos do Dataset V4 com Save-Best

	Dataset V4	
	Sem *Save-Best* (%)	Com *Save-Best* (%)
Screaming	0.00	0.00
Yell	24.69	16.62
Gunshot	12.30	6.88
Music	9.96	3.11
Speech	13.16	6.38
Singing	10.53	7.50
Camera	16.36	10.62
Run	0.00	0.00
Silence	11.76	4.44
Laughter	47.05	50.00
Other	17.80	9.93

Tabela 5.32: Comparação de Save-Best com do Dataset V4

Para concluir, existem duas observações interessantes a mencionar. Primeira, um *dataset* irregular na sua constituição parece obter resultados melhores no teste, do que um *dataset* regular. Como existem mais dados da classe negativa, o modelo parece ser melhor a distinguir entre as classes, o que permite ter uma precisão alta durante o teste. Segunda, ao utilizar um *dataset* regular são obtidos melhores resultados ao utilizar a condição *Save-Best*. Apesar de haver diferenças nos resultados entre os *datasets* V3 e V4, não existe uma diferença significativa, cerca de 1%. Esta diferença também pode ser explicada pelo facto de ter sido utilizado sobreposição na recolha dos vídeos extra. No AudioSet cada troço vem de um vídeo diferente, o que leva a não haver sobreposição, nem semelhança nos dados.

5.5 *Dataset* V5

Uma vez que foram exaustos todos os dados das classes escolhidas presentes no AudioSet, e os resultados obtidos ao utilizar dados fora do AudioSet não resultaram em melhores resultados. O passo seguinte passa por explorar as técnicas de aumentação de dados, previamente descritas. Este *dataset* V5 tem como base o *Dataset* V1, ao qual foi aplicada a rotação de dados, ou seja, cada troço do *dataset* foi rodado 10 vezes, uma vez por cada segundo do troço.

Nome	Classes Positivas		Classes Negativas		Total
	Classes	# de Troços	Classes	# de Troços	
V5	*Screaming*	7580	*Speech*, *Singing*, *Yell*, *Run*	7560 (1890 por classe)	15140

Tabela 5.33: Constituição do Dataset V5, após a aumentação de dados feita com rotação de dados

Esta rotação permite a criação de 10 novos troços, por cada troço no *dataset*. Assim, o número de troços no *Dataset* V5 é 10 vezes maior do que os presentes no *Dataset* V1. Com este *Dataset* pretende-se analisar o resultado da rotação de dados, para verificar se este é de facto útil para a tarefa escolhida. É esperado também que os resultados melhorem.

5.5.1 Avaliação da Experiência YT038

Sabendo que foi feita a rotação de dados sobre o *Dataset* V1, esta experiência é idêntica à primeira experiência desta dissertação, onde não é utilizado o *Save-Best*.

Ao comparar as tabelas 5.34 e 5.2 é possível ver o efeito de adicionar mais dados, e por consequente, o efeito que a rotação de dados teve sobre os troços presentes no *dataset*. A rotação de dados permitiu três coisas. Primeira, ao acrescentar mais dados ao *dataset* resultou em treinos mais eficientes, tendo todas as precisões de treino obtido 100%. Segunda, diminuiu consideravelmente o desvio padrão das métricas de teste, tendo o *Recall* diminuído de 20.44% para 4.80%. Por fim, os valores obtidos ao testar o modelo treinado com o *Dataset* V5 subiram consideravelmente também. A precisão de teste aumentou mais de 10%, passando de 21.68% para 31.84%, o *Recall* passou de 32.42% no YT001 para 40.14% no YT033.

5.5.2 Variação da Experiência YT038

Visto que, os resultados obtidos foram bastante bons, foi repetida a mesma experiência mais 4 vezes. As experiências anteriores demonstraram que os resultados podem variar, devido à inicialização do pesos do modelo ser aleatória.

	Treino	Validação	Teste (Filme)		
ID	Precisão		Precisão	*Recall*	*F-Score*
YT038-a	100	80.20	32.45	46.21	38.12
YT038-b	100	79.04	29.94	35.61	32.53
YT038-c	100	80.64	30.49	37.88	33.78
YT038-d	100	80.66	24.64	38.64	30.09
YT038-e	100	79.44	36.96	38.64	37.78
YT038-f	100	**81.80**	30.27	42.42	35.33
YT038-g	100	80.04	32.09	32.58	32.33
YT038-h	100	80.02	29.17	37.12	32.67
YT038-i	100	80.00	**38.92**	**49.24**	**43.48**
YT038-j	100	80.20	33.53	43.18	37.75
Média	100	80.20	31.84	40.15	35.38
Desvio Padrão	0.0	0.70	3.82	4.80	3.73

Tabela 5.34: Experiência YT038 - Resultados obtidos ao treinar com o *Dataset* V5, todos os resultados em %

ID	Precisão Treino	Precisão Validação	Precisão Teste	*Recall*	*F-Score*
YT038	100	**80.20**	31.84	**40.15**	35.38
YT039	100	79.35	**36.30**	25.15	29.28
YT040	100	78.87	32.54	39.24	35.35
YT041	100	78.92	29.11	34.85	31.34
YT042	100	79.70	35.92	36.59	**36.18**
Média	100	79.41	33.14	35.20	33.51
Desvio Padrão	0.00	0.80	5.83	3.98	2.98

Tabela 5.35: Resultados obtidos ao treinar um conjunto de experiências com o *Dataset* V5, sem *Save-Best*, todos os resultados em %

Ao analisar a tabela 5.35 a primeira coisa que salta a vista é que a precisão do treino é sempre 100%, e que o desvio padrão da precisão de validação é quase nulo. Esta igualdade entre experiências não se verifica durante o teste, onde os desvios padrão aumentaram, apesar de não serem muito altos, tendo a precisão de teste um desvio padrão de 5.83%. Ao comparar esta tabela com a tabela 5.7 é possível ver que, em média, os modelos deste conjunto de experiências são bastante melhores do que os modelos produzidos ao treinar com o *Dataset* V1. Isto deve-se ao número de dados, e por consequente, à rotação de dados, que levou a esse aumento.

Assim, esta primeira análise confirma que é mais eficiente introduzir variações dos dados existentes, do que adicionar dados que não pertençam ao *dataset*. Para além disto, é possível afirmar que o uso de rotação de dados ajudou nessa melhoria do modelo, pois o seu uso não piorou o que se ganharia com o aumento de dados.

5.5.3 Análise de Falsos Positivos

Esta análise é ainda mais interessante devido aos ganhos obtidos pelo conjunto de experiências, que foram treinadas utilizando este novo *dataset*.

	Dataset V1	Dataset V5	
	Média (%)	Média (%)	Desvio Padrão (%)
Screaming	0.00	0.00	0.00
Yell	34.62	17.93	6.23
Gunshot	20.71	4.89	4.80
Music	10.75	5.22	3.85
Speech	21.97	5.29	2.12
Singing	15.18	1.75	4.34
Camera	30.94	19.19	5.66
Run	0.00	0.00	0.00
Silence	13.65	5.11	10.47
Laughter	77.14	74.00	43.86
Other	24.62	11.31	3.98

Tabela 5.36: Comparação dos falsos positivos das experiências treinadas com o *Dataset* V5, e os das experiências treinadas com *Dataset* V1

A diferença entre os resultados conseguidos ao utilizar o *Dataset* V5, em vez do *Dataset* V1 é bastante clara, ao analisar a tabela 5.36. Em todas as *labels* houve um descida clara no número médio de falsos positivos, sendo o mais importante a descida na etiqueta *Speech*, pois esta é a que tem mais segmentos no filme. Esta diminuição confirma os pontos observados em cima - ao adicionar mais dados ao *dataset*, melhor é o modelo treinado com esse *dataset*; e a rotação de dados não é um factor limitante nos resultados obtidos.

5.5.4 Avaliação da Experiência YT043

Novamente serão feitas mais experiências utilizando a condição *Save-Best*. Como nos *Dataset* anteriores, é esperado que, ao utilizar o *Save-Best*, o modelo melhore.

Ao analisar a tabela 5.37, e comparar com a tabela 5.35, é possível ver que, apesar da precisão ter melhorado, passando de 31.84% na experiência YT033 para 32.79% no YT035, o *F-Score* diminuiu consideravelmente, passando de 35.45% para 30.93%. Isto foi devido à descida do *Recall*, que passou de 40.15% no YT033 para 30.53%, uma descida de quase 10%. Isto significa que, o modelo que não foi treinado sobre a condição *Save-Best*, é melhor a identificar os segmentos positivos do filme.

	Treino	Validação	Teste (Filme)		
ID	Precisão		Precisão	*Recall*	*F-Score*
YT043-a	100	80.10	**43.48**	22.73	29.85
YT043-b	100	79.22	36.27	28.03	31.62
YT043-c	100	80.66	32.12	33.33	32.71
YT043-d	100	79.14	26.32	30.30	28.17
YT043-e	100	80.40	39.56	27.27	32.29
YT043-f	100	79.62	36.44	32.58	34.40
YT043-g	100	**80.76**	32.24	**37.12**	**34.51**
YT043-h	100	80.10	24.59	34.09	28.57
YT043-i	100	80.28	37.04	30.30	33.33
YT043-j	100	80.02	19.90	29.55	23.78
Média	100	80.03	32.79	30.53	30.92
Desvio Padrão	0	0.52	6.93	3.83	3.19

Tabela 5.37: Experiência YT035 - Resultados obtidos ao treinar com o *Dataset* V5, sobre a condição *Save-Best*, todos os resultados em %

5.5.5 Variação da Experiência YT043

Como os resultados obtidos na experiência YT043, não foram óptimos, foi repetida mais quatro vezes a experiência.

	Precisão Treino	Precisão Validação	Precisão Teste	*Recall*	*F-Score*
YT043	100	**80.03**	32.79	30.53	30.92
YT044	100	79.24	28.20	31.29	28.47
YT045	100	78.41	32.42	35.30	32.66
YT046	100	79.15	30.14	31.06	29.36
YT047	100	79.97	**33.49**	**37.95**	**34.26**
Média	100	79.36	31.41	33.23	31.13
Desvio Padrão	0.00	0.59	7.41	8.19	5.22

Tabela 5.38: Resultados obtidos ao treinar um conjunto de experiências com o *Dataset* V5, com *Save-Best*, todos os resultados em %

Ao comparar a tabela 5.38 com a tabela 5.35, é fácil perceber que os resultados obtidos pelo novo conjunto de experiências, onde foi utilizado a condição *Save-Best*, são piores. Pois, houve uma diminuição de todas as métricas de teste. No total, o *F-Score* diminuiu de 33.13% para 31.13%. Apesar de ter diminuído pouco (2%), é uma diminuição, algo que não é esperado.

Esta diferença, entre o uso ou não uso do *Save-Best*, é inesperada, e pode ser devida ao conjunto de validação utilizado. Como o conjunto de validação é criado com dados diferentes aos de teste, é possível que estes não sejam os correctos para validar o modelo. Assim, um conjunto de validação, com base no teste, seria mais apropriado.

5.5.6 Análise de Falsos Positivos

Dado que, a precisão de teste diminuiu, ao usar o *Save-Best*, é esperado que os falsos positivos aumentem.

	Sem *Save-Best*	Com *Save-Best*	
	Média (%)	Média (%)	Desvio Padrão (%)
Screaming	0.00	0.00	0.00
Yell	17.93	19.63	7.56
Gunshot	4.89	6.55	6.82
Music	5.22	6.75	7.80
Speech	5.29	6.38	5.42
Singing	1.75	5.25	8.32
Camera	19.19	19.75	7.30
Run	0.00	0.00	0.00
Silence	5.11	5.11	11.15
Laughter	74.00	78.00	41.42
Other	11.31	11.76	7.57

Tabela 5.39: Comparação do efeito que o uso da condição *Save-Best* tem sobre os falsos positivos, ao treinar um modelo com o *Dataset* V5

Assim, como ilustra a tabela 5.39, é possível ver o efeito que o uso do *Save-Best* teve, sobre os modelos treinados com o *Dataset* V5. Sendo que, a diferença entre as precisão de teste dos dois conjuntos de experiências não é grande, o aumento do número médio dos segmentos mal classificados também não é grande.

5.6 *Dataset* V6

Como os resultados obtidos com o *Dataset* V5 foram promissores, este *dataset* foi construído para estudar o efeito de outro método de aumentação de dados - o controlo de volume. Para a construção do *Dataset* V6 foi utilizada a mesma constituição do *dataset* V1, ao qual foi alterado o volume.

Nome	Classes Positivas		Classes Negativas		Total
	Classes	# de Troços	Classes	# de Troços	
V6	Screaming	3790	Speech, Singing, Yell, Run	3788 (947 por classe)	7578

Tabela 5.40: Composição do *Dataset* V6, após a aumentação de dados com controlo de volume

Antes de fazer o *dataset*, foram analisados todos os troços presentes no *Dataset* V1, e retirado o valor máximo, mínimo e a mediana dos valores das *features* dos espectrogramas dos mesmos. Com estes valores, foram criadas regras heurísticas para controlar em que troços o volume seria aumentado, e em que troços o volume seria diminuído. Se a mediana do espectrograma de um troço, se encontra a cima da mediana do *dataset*, esse troço será diminuído. Caso contrário, será aumentado. Com estas regras, foi feito o controlo de volume aos troços utilizados no *dataset*.

Dado que, este tipo de aumentação de dados, não parece produzir variação, considerada suficiente, nas *features* do espectograma, não é esperado que os modelos treinados com este *dataset*, produzam melhores resultados do que os treinados pelo *Dataset* V5. Assim, existe a possibilidade de os resultados do teste piorarem, em relação às experiências anteriores. Mas, sendo que foram adicionados mais dados, continua a ser esperado que, ao aumentar o número de dados, os resultados melhorem.

5.6.1 Avaliação da Experiência YT048

Para esta primeira experiência com o *Dataset* V6 não foi utilizado a condição *Save-Best*. Poderá então ser utilizado o conjunto de experiências, onde se insere a experiência YT038, do *Dataset* V5, pois ambas foram feitas sobre as mesmas condições. Assim, será possível analisar qual dos métodos produz melhores resultados.

Numa comparação directa, entre a tabela 5.41 e a tabela 5.34, pode-se verificar que o método de controlo de volume produz piores resultados, ao ser testado com o filme, que a rotação de dados. Entre as primeiras experiências, existe uma diferença clara, tendo a rotação de dados obtido 35.38% no *F-Score*, e o controlo de volume 27.38%. Outro ponto de consideração é a precisão de teste, onde a experiência YT038 obteve 31.84% e a experiência YT048 obteve 20.63%, uma diminuição de mais de 10%. Por fim, no *Recall*,

	Treino	Validação	Teste (Filme)		
ID	Precisão		Precisão	*Recall*	*F-Score*
YT048-a	100	75.24	16.48	45.45	24.19
YT048-b	100	77.44	22.97	38.64	28.81
YT048-c	100	75.84	15.80	**53.03**	24.35
YT048-d	100	75.64	**26.23**	48.48	**34.04**
YT048-e	100	75.92	26.04	37.88	30.86
YT048-f	100	74.52	19.31	42.42	26.54
YT048-g	100	76.32	23.98	35.61	28.66
YT048-h	100	**78.24**	12.74	40.15	19.34
YT048-i	100	77.40	21.70	38.64	27.79
YT048-j	100	75.00	21.00	47.73	29.17
Média	100	76.16	20.63	42.80	27.38
Desvio Padrão	0.00	1.13	4.28	5.35	3.85

Tabela 5.41: Resultado das execuções da experiência YT048, utilizando o *Dataset* V6, sem *Save-Best*, todos os resultados em %

o contrário aconteceu, onde o YT038 obteve 40.15%, e o YT048 teve 42.8%. Assim, é possível dizer que, os modelos produzidos para a experiência YT045 são melhores para identificar correctamente os segmentos do filme que são positivos. Ao mesmo tempo, a experiência que utilizou o *Dataset* V6 (YT048) tem desvios padrão ligeiramente mais elevados.

5.6.2 Variação da Experiência YT048

Os resultados obtidos na experiência YT048 demonstram que, os modelos treinados com o *Dataset* V6 são piores em média, do que os treinados com o *Dataset* V5. Aliás, este resultado é ainda mais preocupante quando comparamos com os resultados da experiência YT001. Ao comparar a tabela 5.41 com a tabela 5.2 é possível ver que a precisão baixou levemente, passando de 21.68% na experiência YT001, para 20.63% na experiência YT048. Apesar disto, houve um aumento considerável no *Recall*, cerca de 10%, levando assim, a um aumento do *F-Score*. Para confirmar estes valores, foram executadas mais quatro experiências.

Ao analisar a tabela 5.42, e compara-la com a tabela 5.35, a distinção entre os dois métodos de aumentação de dados é clara, o método de controlo de volume não oferece melhores resultados. Para começar, ao utilizar o *Dataset* V6 são obtidos piores resultados em precisão e *F-Score* no teste. O resultado da precisão baixou de 33.14% no conjunto de experiências do YT038, para 22.36%. Por isto, o valor do *F-Score* diminuiu de 33.51% para 28.31%. Apesar disto, os modelos treinados com o *Dataset* V6 são melhores, a identificar correctamente os troços positivos, que os modelos treinados pelo *Dataset* V5, tendo o *Recall* aumentado de 35.20% para 40.91%.

ID	Precisão Treino	Precisão Validação	Precisão Teste	*Recall*	*F-Score*
YT048	100	76.16	20.63	42.80	27.38
YT049	100	**77.17**	**24.57**	38.33	**29.56**
YT050	100	75.88	21.46	**43.26**	28.03
YT051	100	76.84	22.07	37.95	27.53
YT052	100	76.76	23.07	42.20	29.06
Média	100	76.56	22.36	40.91	28.31
Desvio Padrão	0	0.87	4.07	7.94	2.62

Tabela 5.42: Experiências feitas com o *Dataset* V6, com controlo de volume, todos os resultados em %

Estes resultados eram, de certa maneira esperados, tendo em conta a falta de variação. O que não era esperado era a melhoria dos modelos treinados com o *Dataset* V6, em identificar os segmentos positivos do filme.

5.6.3 Análise de Falsos Positivos

Sendo que a precisão dos modelos do *Dataset* V6 piorou, e o *Recall* aumentou, é esperado que os falsos positivos, em relação aos obtidos pelo conjunto de experiências treinadas com o *Dataset* V5, sem *Save-Best*, aumentem em todas as etiquetas.

	Dataset V5	Dataset V6	
	Média (%)	Média (%)	Desvio Padrão (%)
Screaming	0.00	0.00	0.00
Yell	17.93	30.55	7.70
Gunshot	4.89	17.55	8.04
Music	5.22	5.88	4.37
Speech	5.29	15.84	6.67
Singing	1.75	10.50	9.13
Camera	19.19	26.81	13.23
Run	0.00	0.00	0.00
Silence	5.11	3.55	6.82
Laughter	74.00	96.00	19.59
Other	11.31	19.01	7.27

Tabela 5.43: Comparação entre os falsos positivos obtidos pelo *Dataset* V5 e os obtidos pelo *Dataset* V6

E de facto, isso aconteceu. Realmente, em todas as etiquetas (except o *Silence*), os modelos que foram treinados pelo *Dataset* V6, são piores a identificar a classe negativa. É de notar a etiqueta *Speech*, sendo que é a etiqueta com mais segmentos no filme, aumentou mais de 10% no número de segmentos incorrectamente classificados.

5.6.4 Avaliação da Experiência YT053

Para confirmar que, ao utilizar o controlo de volume como método de aumentação de dados, os resultados dos testes destes modelos piora, foi produzido um conjunto de experiências, que desta vez, será feito sobre regime *Save-Best*.

	Treino	Validação	Teste (Filme)		
ID	Precisão		Precisão	*Recall*	*F-Score*
YT053-a	100	76.84	18.70	32.58	23.76
YT053-b	100	75.84	25.70	34.85	29.58
YT053-c	100	75.28	12.97	23.48	16.71
YT053-d	100	76.00	**33.12**	39.39	**35.99**
YT053-e	100	**77.08**	19.90	28.79	23.53
YT053-f	100	77.04	23.56	31.06	26.80
YT053-g	100	76.80	15.81	25.76	19.60
YT053-h	100	73.32	14.98	**49.24**	22.97
YT053-i	100	76.84	26.78	37.12	31.11
YT053-j	100	75.36	14.23	28.03	18.88
Média	100	76.04	20.57	33.03	24.89
Desvio Padrão	0	1.11	6.22	7.17	5.7

Tabela 5.44: Resultados das diferentes execuções da Experiência YT053, todos os resultados em %

Ao comparar as tabelas 5.44 e 5.42, é possível ver o efeito que o *Save-Best* tem sobre este *dataset*. Apesar da precisão ter-se mantido praticamente no mesmo, houve uma diminuição grave do *Recall*, que passou de 42.80% para 33.03%. Assim, os modelos treinados com o *Dataset* V6, com o *Save-Best*, são piores a identificar os segmentos positivos. Mas, dado que a precisão manteve-se, isso significa que este modelos, são melhores a identificar correctamente os negativos, o que levará a uma diminuição dos falsos positivos.

Quando é comparado as tabelas 5.44 e 5.37, é observado que o modelo treinado com o *Dataset* V6 é melhor a identificar correctamente os positivos, tendo o *Recall* melhorado de 30.53% no *Dataset* V5, para 33.03%. Mas, ao todo, estes modelos são consideravelmente piores, pois existe uma diferença grande na precisão, que leva a um diferença no *F-Score*. A precisão dos modelos treinados pelo *Dataset* V5, sobre *Save-Best* é de 32.79%, enquanto o equivalente do *Dataset* V6 é de 20.57%. Isto leva a que, o *F-Score* diminua de 30.92% para 24.89%.

5.6.5 Variação da Experiência YT053

Como nos *datasets* anteriores, foi repetida a experiência YT048 mais quatro vezes. Sendo que, os resultados da experiência YT048 não revelaram melhorias em relação ao conjunto

de experiências do YT043, e, em especial, aos conjuntos de experiências do *Datset* V5, é expectável que, apesar da repetição, os resultados do teste não melhorem.

ID	Precisão Treino	Precisão Validação	Precisão Teste	*Recall*	*F-Score*
YT053	100	76.04	20.57	33.03	24.89
YT054	100	**77.19**	**23.76**	40.53	**28.67**
YT055	100	75.10	21.95	35.98	25.74
YT056	100	77.09	15.49	**41.89**	22.17
YT057	100	76.83	20.81	40.45	26.98
Média	100	76.45	20.51	38.38	25.69
Desvio Padrão	0.00	1.26	3.39	6.14	3.06

Tabela 5.45: Experiências feitas com o *Dataset* V6, com controlo de volume, sobre *Save-Best*, todos os resultados em %

Tal como esperado, os resultados do conjunto de experiências do YT048, revelaram valores semelhantes. Ao comparar a tabela 5.45 com a tabela 5.42, é possivel observar que não houve mudanças drásticas. O conjunto de experiências, que não foi treinado com *Save-Best*, produziu em todas as métricas de teste, uma melhoria de cerca de 2%. Para além disto, ao comparar as tabelas 5.45 e 5.38, a diferença é semelhante à verificada, ao comparar os conjuntos de experiências, feitos sem *Save-Best*. É assim possível dizer que, apesar de ser adicionado mais dados ao *dataset* de treino, e por sua vez, ao de validação, os dados adicionados não produzem melhores modelos.

5.6.6 Análise de Falsos Positivos

Apesar de terem sido obtidos piores resultados, a análise de falsos positivos continua a ser importante, visto que, como foi dito na avaliação do YT053, os modelos produzidos sobre regime de *Save-Best*, poderam ser melhores a correctamente identificar os segmentos negativos presentes no teste.

Apesar de ter havido uma melhoria no YT048, ao fazer mais quatro experiências, essas melhorias param de existir. Assim, em praticamente todas as etiquetas, não existe diferença na percentagem média de falsos positivos.

	Sem Save-Best	Com Save-Best	
	Média (%)	Média (%)	Desvio Padrão (%)
Screaming	0.0	0.00	0.00
Yell	30.55	30.91	10.23
Gunshot	17.55	18.22	9.09
Music	5.88	7.33	5.99
Speech	15.84	17.51	9.33
Singing	10.50	12.00	10.88
Camera	26.81	26.75	12.81
Run	0.00	0.00	0.00
Silence	3.55	7.55	15.78
Laughter	96.00	90.00	30.00
Other	19.01	20.53	9.84

Tabela 5.46: Comparação que o uso do *Save-Best* tem, quando treinado com o *Dataset* V6

5.7 Análise Gráfica da Classificação do Melhor Modelo

Após terem sido feitas todas as experiências, foi feita uma análise gráfica das classificações obtidas pelo melhor modelo. Para a escolha do melhor modelo, foi apenas considerado o conjunto de experiências que obteve melhores resultados. Assim, foi feita uma comparação dos *F-Scores* de todas as execuções nesse conjunto de experiências. A execução escolhida foi a 8ª execução da experiência YT035, que teve um *F-Score* de 44.19%.

Na figura 5.1, o eixo vertical representa o *Ground Truth*, ou seja, a classe verdadeira, e o eixo horizontal representa a classe que o modelo classificou. Assim, é possível ver que, este modelo classificou correctamente 47.82% de todos os segmentos negativos. Ao mesmo tempo, apenas 22.03% dos segmentos positivos foram correctamente classificados. Com este gráfico, é possível observar a precisão do modelo.

Na página seguinte, apresentam-se duas figuras 5.2 e 5.3 que permitem visualizar a precisão, e a certeza do modelo, ao longo de filme.

Nestas figuras, o canto superior esquerdo representa o primeiro segmento do filme, e a última caixa da linha 20, representa o último segmento do filme. A figura 5.2 representa a certeza do modelo para cada um dos segmentos do filme. Nesta figura, quando mais escuro o azul, mais certo o modelo está que o segmento pertence à classe positiva. Quanto mais branco, mais certo está que o segmento pertence à classe negativa. A figura 5.3 representa o *Groud Truth*, onde os segmentos a azul escuro representam os segmentos do filme que pertencem à classe positiva, enquanto os a branco à classe negativa.

Como é possível ver, existem muito poucas ocasiões em que o modelo está bastante certo da sua classificação.

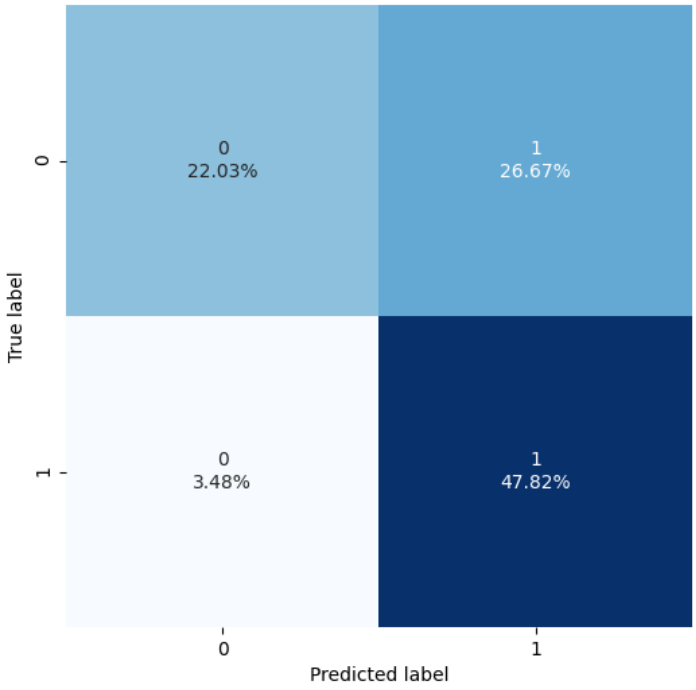

Figura 5.1: Matriz de Confusão das classificações feitas pela execução, sobre o filme

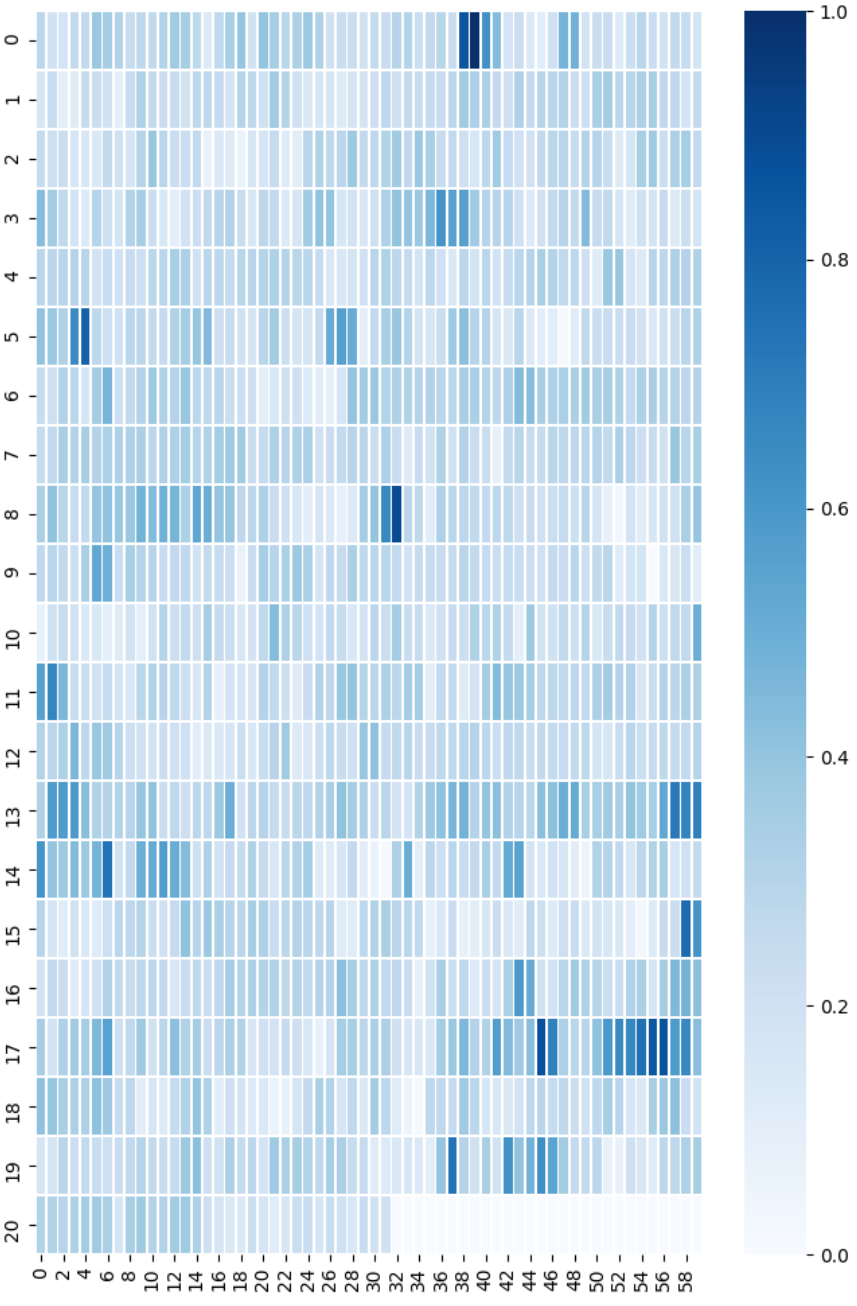

Figura 5.2: Certeza da classificação de cada segmento do filme.

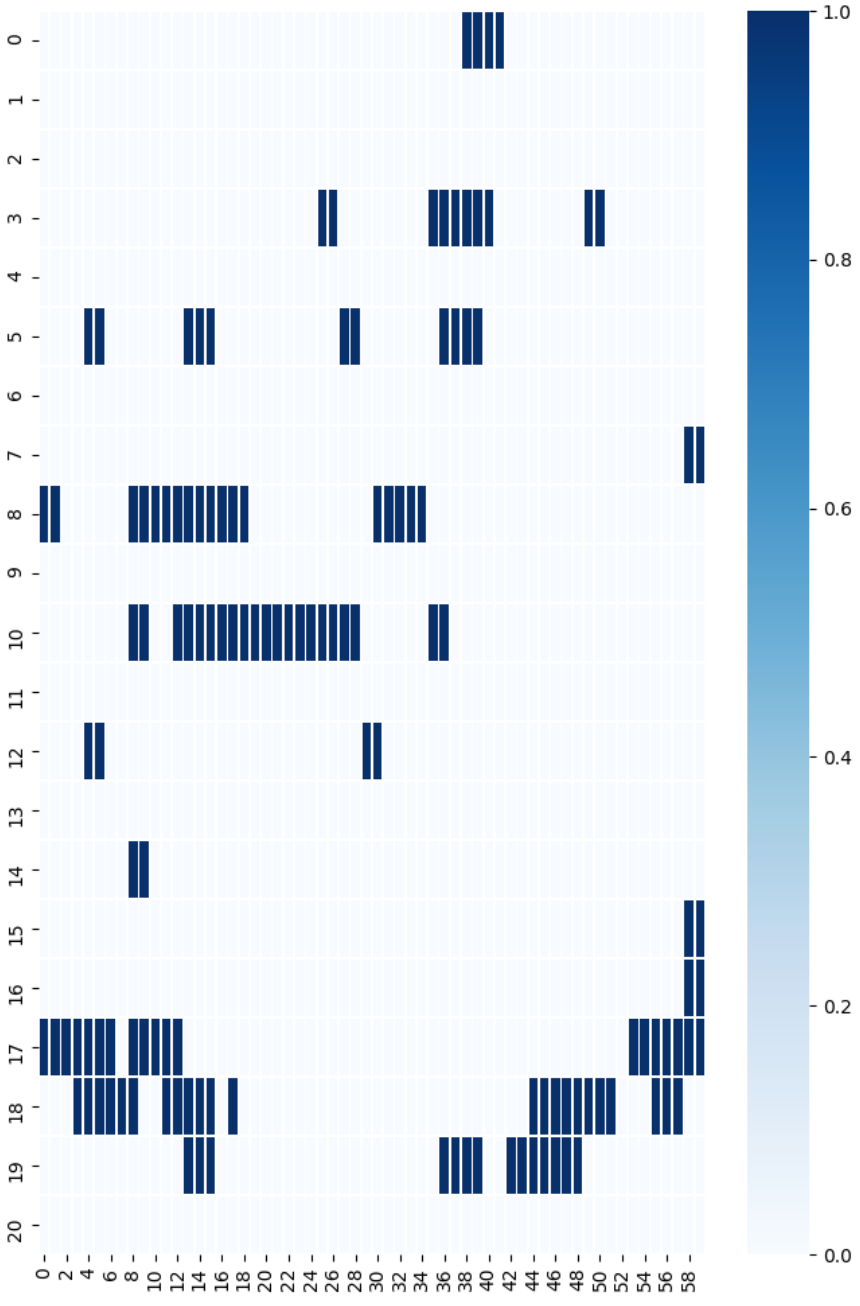

Figura 5.3: Matriz de Confusão das classificações feitas pela execução, sobre o filme

Capítulo 6

Conclusão

A Inteligência Artificial é cada vez mais importante no dia-a-dia e o número de estudos e investigações aumentaram bastante nos últimos anos, e continuarão a aumentar. Esta dissertação também pretende contribuir para esse estudo. A classe de modelos de AI utilizado foi Rede Neural, mais especificamente o modelo VGG, que serviu como base para a arquitectura do modelo utilizado. Para treinar o modelo, foi necessário, identificar um *dataset* apropriado, e depois analisa-lo. O que se verificou é que, primeiro, haviam poucos *datasets* relevantes e realmente bons para o estudo e, segundo, poucas manipulações e conclusões feitas à cerca dos mesmos. O estudo dos *datasets* sonoros poderão ajudar, no futuro, a identificar as emoções sentidas pelas pessoas através da análise da fala.

Os *datasets* estudados foram AMIGOS, DEAM, EMOMUSIC e AudioSet. Sendo que, os três primeiros foram descartados por apresentarem poucos dados, falta de coerência e falta de validações, respectivamente. Nenhum dos três não enquadrou-se nos pré-requisitos para um *dataset* neste trabalho. Deste modo, foi feito um pivôt para o AudioSet, um *dataset* composto de vídeos do YouTube, anotados e validados. Com este *dataset* foi também feito um pivôt no objectivo do trabalho, passando da criação de um modelo para emoções explicitas, para a criação de um modelo que classifique emoções implícitas. O uso deste *dataset* trouxe, também, uma maior facilidade na análise e validação de resultados, visto que estes não são subjectivos, isto é, não variam de pessoa para pessoa (um som de um grito é um grito, independentemente da emoção).

O primeiro *dataset* criado - V1 - era composto por 1 etiqueta de classe positiva, e por 4 de classe negativa, permitindo estudar diversas variações no treino do modelo. Dos resultados obtidos no V1, foi possível concluir que, pode existir uma melhoria de resultados de teste ao utilizar a condição *Save-Best*, que guarda sempre o melhor modelo. Com treino do modelo sem validação, foi possível ver que, ao utilizar mais dados no treino, já não era necessário utilizar os dados para a validação. Isto implicou haver mais dados para o treino, obtendo melhores resultados no teste. Ao mesmo tempo, a validação obtém resultados mais estáveis entre as diferentes execuções da mesma experiência.

Estas experiências com *Dataset* V1 ajudaram a definir as próximas validações e testes

que serão efectuados nos *datasets* seguintes. Assim, para todos os *datasets* estudados, foi sempre utilizado um conjunto de validação, e foram feitos dois conjuntos de experiências - um com *Save-Best* e um sem. Outro ponto importante analisado em todos os *datasets* foram os falsos positivos. Isto permitiu verificar se o modelo está a ficar mais ou menos correcto a identificar os dados negativos. Neste trabalho, o conceito de falso positivo refere-se aos segmentos do filme que deviam estar na classe negativa, mas foram classificados como positiva.

Ao analisar os falsos positivos, do *Dataset* V1 foi possível observar que uma das *labels* que ficou mais mal classificada era o som de camera fotográfica. Assim, no *Dataset* V2, introduziu-se esta *label* na classe negativa, na expectativa de diminuir os falsos positivos. Para construção do *Dataset* V2, foi mantida a mesma repartição de dados do V1, isto é, ter o mesmo número de troços da classe negativa e positiva, diminuindo o número de troços por cada etiqueta da classe negativa. Após o primeiro conjunto de experiências (sem *Save-Best*), o número de falsos positivos da etiqueta em questão (*Camera*) diminuiu para metade. Ao fazer o mesmo conjunto de experiências, com *Save-Best*, são obtidos ainda melhores resultados, tanto nas métricas de teste, bem como na identificação correcta da *label*. O *Dataset* V2 permitiu concluir que, aumentar o número de *labels* na classe negativa, pode resultar na melhoria dos resultados. No entanto, é de notar que, não se recomenda repartir demasiado as *labels*, pois pode resultar em poucos troços por *label*, não permitindo uma aprendizagem correcta da classe negativa.

Para perceber o efeito causado pelo *dataset* irregular, foi construído o V3. Este, ao contrário do V2, não manteve o mesmo número de repartições, mas sim, o número de dados por etiqueta. Isto resultou num aumento de troços da classe negativa, enquanto a positiva manteve-se. Os resultados foram contrários do esperado, isto é, ao utilizar o *Save-Best* deveriam ter se obtidos melhores resultados, algo que aqui não aconteceu. Mas, os resultados do *Save-Best* foram, ao todo, considerados mais estáveis, pois tinham um menor desvio padrão. Acredita-se que este tipo de *datasets* podem levar a resultados imprevisíveis, algo que não é benéfico a qualquer estudo científico.

Com o *Dataset* V4, pretendeu-se perceber se, aumentando o número de dados da classe positiva poderíamos obter melhores resultados tanto no treino e validação, como também depois com o teste do filme. Para suplementar a classe positiva, de modo a que tenha o mesmo número de dados que a negativa do V3, foram utilizados vídeos com compilações de efeitos sonoros. Ao utilizar este *dataset* para o treino do modelo, os resultados obtidos com *Save-Best* foram melhores do que sem essa condição, mas no geral os desvios padrão foram piores que o V3. Isto pode implicar que, mesmo que, as classes estejam equilibradas, como se utilizaram dados fora do *dataset* (sem conhecimento da sua qualidade), isso pode levar o modelo a identificar pior os segmentos de teste. Neste *dataset* também houve aumento dos falsos positivos, isso deve-se ao facto dos dados acrescentados piorarem a identificação correcta da classe positiva.

Apesar AudioSet ter mais de 2 milhões de vídeos anotados, é composto de imensas *labels*, e, por isso, cada classe pode ter poucos troços. A *label* em estudo, *Screaming*, tinha apenas 758 troços no *dataset*. Como o V3 e V4 não obtiveram resultados melhores que o V2, foram estudadas técnicas de aumentação de dados. As técnicas de aumentação de dados, permitem a criação de novos dados a partir dos dados originais, fazendo algum tipo de manipulação sobre estes. As duas técnicas estudadas foram: rotação de dados e controlo de volume.

Para o *Dataset* V5 foi utilizada a primeira técnica de aumentação de dados: rotação de dados. A constituição deste *dataset* foi igual ao do *Dataset* V1. Após o primeiro conjunto de experiências, foi possível ver o efeito positivo, que a adição de mais dados teve, sobre os modelos, pois estes obtiveram aumentos nas percentagens das métricas de teste. Apesar deste efeito positivo, ao utilizar o *Save-Best*, não foi conseguida a melhoria esperada. Supõem-se que, o problema encontra-se na validação. Com tantos dados semelhantes, é possível que, durante a validação, na condição *Save-Best*, os dados utilizados não tenha diferenças suficientes para uma validação correcta. Também se suspeita que, ao utilizar dados fora do *dataset* possa melhorar este processo.

Tal como o *Dataset* V5, o V6 foi construído com a mesma constituição do V1, mas utilizando o controlo de volume. Ao usar esta técnica, existe uma diferença notável no volume dos áudio, mas ao criar o espectograma, essas diferenças são mínimas. Ao comparar os resultados obtidos, pelos modelos treinados com o *Datasets* V5 e V6, é claro que, os modelos treinados pelo *Dataset* V6, seja ou não com a condição *Save-Best*, são consideravelmente piores. Conclui-se aqui que, entre as duas técnicas de aumentação de dados, a que resultou melhor foi a rotação de dados.

Estes *datasets* foram uma tentativa de simular um *dataset* aberto, ou seja, um *dataset* que possa ser alterado ao longo do tempo. Em primeiro, verificou-se que, a validação é essencial para haver melhores resultados de teste. Em segundo, aumentar as etiquetas pode também melhorar resultados. Contudo, é preciso ter atenção ao número de dados por etiqueta em cada classe. Em terceiro, utilizar dados fora do *dataset* (de pouca confiança), não é uma prática recomendada, tal como, a utilização de *datasets* irregulares. Por último, na situação de poucos dados, recomendam-se as técnicas de aumentação dos dados existentes. Sendo que, a melhor aparenta ser a de rotação de dados.

Este trabalho poderá ser relevante no estudo das técnicas de aumentação de dados, *datasets* abertos e outras técnicas alternativas no treino de modelos. No futuro poderão ser realizadas mais experiências, como por exemplo, aumentar as classes positivas, combinar as duas técnicas de aumentação de dados e também o estudo da introdução de ruído.

Apêndice A

Código

Todo o código utilizado nesta dissertação está presente neste anexo. Apenas será mostrado o nome das funções, bem como a documentação da mesma.

A.1 Scripts

A.1.1 Audioset

Script que cria os datasets com base no Audioset.

Utiliza as seguintes funções:

- build_directories
- build_base_dataset
- new_download_procedure
- cut
- data_rotation
- avg_length_by_folder
- get_downloaded_videos
- create_dataset

Este script utiliza duas listas: uma para as classes positivas e outra para as classes negativas, retira os IDs dos vídeos que correspondem a essas classes, faz o download (se ainda não foi feito), retira o áudio, corta o troço de áudio indicado pelo dataset, faz o downsampling desse troço, e cria os datasets - treino e teste.

new_download_procedure

```
    Procedimento de Download.

    Args:
        id: ID do vídeo a ser feito download.
        start: Segundo de começo do troço a ser retirado do vídeo.
        end: Segundo de fim do troço a ser retirado do vídeo.
        classe: \textit{Label} a que pertence o vídeo.
```

extra_download

```
    Faz o download de vídeos que não pertencem ao AudioSet

    Args:
        id: ID do vídeo a ser feito download.
        destFolder: Directoria onde será guardado o vídeo.
        cutSize: Tamanho da janela de corte.
        overlap: \textit{Shift} em segundos da janela de corte.
```

avg_length_by_folder

```
    Duração médio dos vídeos numa dada directoria

    Args:
        folder: directoria com vídeos.

    Returns:
        float: duração média dos vídeos
```

get_downloaded_videos

```
    Lê a directoria de download, e cria um
    dicionário com a constituição do mesmo.

    Args:
        folder: directoria composta de directorias com vídeos.

    Returns:
        dict: Dicionário dos vídeos que foram feitos download.
```

pre_dataset

```
Cria as listas que serão utilizadas para a construção do
dataset

Args:
    download_folder: Directoria para onde os vídeos
        foram feitos download
    equal_split: Define se o \textit{split} entre as
        classes positiva e negativo é igual
    split: Define o tamanho do \textit{split}
        entre as duas classes.
    extra: Define se vai ser utilizado vídeos
        fora do AudioSet
    extra_pos: Classes positivas extra

Returns:
    list: Composta por tuplos, onde o primeiro elemento
        é o caminho para o troço do vídeo
        e o segundo a \textit{label}.
```

pre_dataset_with_file

```
Cria as listas que serão utilizadas para a construção do
dataset, utilizando ficheiros com a ordem de outro
dataset, criados durante a  construção.

Args:
    path: Caminho para a directoria onde
        os vídeos se encontram.
    train: Caminho para o ficheiro com os troços
        que pertencem ao treino
    test: Caminho para o ficheiro com os troços
        que pertencem ao teste

Returns:
    List, List: A primeira lista contém os caminhos
    para os troços e labels que constituem o dataset
    de treino, e a segunda a de teste.
```

A.2 DatasetTools

Todas as ferramentas necessárias para a construção e verificação de *datasets*, para o download e tratamento de vídeos e para o tratamento de ficheiros áudio.

A.2.1 DatasetTools

convert_labels

```
Converte as labels do AudioSet
que estão em Unicode, para texto.

Args:
    pos_classes: Lista com as labels da classe positiva
    neg_classes: Lista com as labels da classe negativa
    file: Caminho para o ficheiro do Audioset com as labels.

Returns:
    Dict: Dicionário com as todas as conversões
    List: Lista com as labels positivas convertidas
    List: Lista com as labels negativas convertidas
```

build_base_dataset

```
Cria uma pré-dataset que será utilizado para o download.

Args:
    positive_labels: Lista das labels
        que constituem a classe positiva
    negative_labels: Lista das labels
        que constituem a classe negativa

Returns:
    Dict: Dicionário que será utilizado para o download.
```

write_dataset_json

```
Escreve o pré-dataset para um ficheiro json.

Args:
    dataset: Pré-Dataset
    filePath: Caminho onde será guardado este ficheiro.
```

read_dataset_json

```
    Lê o ficheiro JSON com o pré-dataset.

    Args:
        filePath (str): Caminho para o ficheiro dataset.

    Returns:
        Dict: Pré-Dataset
```

clean_labels

```
    Faz a limpeza das strings
        com as labels presentes no AudioSet

    Args:
        conv: Caminho para o ficheiro dataset.
        labels:
    Returns:
        List: Pré-Dataset
```

create_dataset

```
    Esta função lê os áudios, converte para espectogramas,
    e escreve os espectograms e as respectivas labels para
    os seus ficheiros, assim criando os datasets.

    Chama:
        - stftFeatures.writeStft8bitsFiles

    Args:
        data: Lista com os caminhos para os áudios
        labels: Lista com a labels que correspondem
            aos áudios
        dataname: Caminho para p ficheiro de dados
        labelname: Caminho para o ficheiro de labels
        height: Altura esperada do espectograma.
        width: Largura esperada do espectograma.
```

read_labels

```
Lê e retorna um ficheiro de labels.

Args:
    path: Caminho para o ficheiro de labels.

Returns:
    List: Ficheiro de labels.
```

build_directories

```
Função que cria as directorias base,
    se não existirem.

Args:
    basePath: Caminho para onde serão criadas
        as directorias
    labels: Lista com as labels das classes.
```

A.2.2 AudioTools

full_link

```
Retorna o link do youtube utilizado para
    fazer o download.

Args:
    id: ID do video.

Returns:
    str: Link do Youtube.
```

download_songs

```
Função responsável pelo download do vídeo.
Converte automaticamente o vídeo em áudio

Args:
    downloadFolder: Caminho para a directoria
        onde o áudio será guardado.
    d: link do vídeo do Youtube.
```

list_download

```
    Função responsável pelo o download de vários vídeos.
    Recebe uma lista com IDs de vídeos, e chama a função
        download_song para que seja feito o download.

    Args:
        idList: Lista com IDs.
        downloadFolder: Caminho da directoria onde
            será guardado os áudios da lista.
```

data_rotation

```
    Função que trata da rotação do áudio numa dada
    directoria. Roda dataLength vezes os dados.
    Normalmente, dataLength deverá igualar ao tamanho dos
    troços de áudio - 10 para 10 segundos.

    Args:
        download_folder: Caminho para a directoria onde se
            encontram todos as classes
            que foram feitas download.
        classe: Classe desejada para o download
        dataLength: Número de rotações feitas com o troço
            de áudio.
```

cut_audio

```
    Função encarregue de fazer o corte do áudio.

    Args:
        start: Segundo de começo do corte
        end : Seundo de fim do corte
        filepath: Caminho para o áudio que será cortado
        length: Dimensão esperada. Defaulta para None.
        outputFilename: Nome com que será guardado o troço
            cortado. Se for None, o troço é guardado
            com o mesmo nome.
        destFolder: Caminho onde será guardado o troço
            cortado. Se for None, o troço é guardado
            no mesmo sítio.
```

remove_uncut

```
Esta função apaga todos os troços que não
    foram cortados.

Args:
    downloadFolder: Caminho para a directoria de corte
    length: Dimensão esperada após o corte
```

remove_short

```
Retira qualquer áudio cuja dimensão esteja
    abaixo do esperado.

Args:
    downloadFolder: Caminho para a diretoria.
    length: Dimensão esperada do áudio.
```

extend_audio

```
Chamada da função que estica um troço de áudio.
Chama:
    - audioModule.extend_audio

Args:
    filepath: Caminho para o troço
        a ser esticado
    length: Dimensão esperada.
```

extend_audio_folder

```
Esta função estica uma directoria inteira.
Chama:
    - extend_audio

Args:
    downloadFolder: Caminho para a directoria.
    length: Dimensão esperada
```

rename_cut_audio

```
Funcão que altera o nome de audios cortados.
```

```
    Args:
        folder: Caminho para a Directoria.
```

downsampling

```
    Função que faz o downsampling de um áudio.

    Args:
        inputPath: Caminho para o áudio
        outputPath: Caminho onde o áudio vai ser guardado
        rate: Bitrate para o downsampling
```

downsample_folder

```
    Função que faz o donwsampling de uma directoria.
    Chama:
        - downsampling
        - rename_downsampling

    Args:
        folderPath: Caminho para a directoria.
        rate: Bitrate esperado.
```

rename_downsample

```
    Função que muda o nome de um troço que foi
        feito o downsampling

    Args:
        path: Caminho para a directoria
        rate: Bitrate do downsampling
```

volume_changer

```
    Altera o volume de um áudio

    Args:
        input_file: Caminho para a directoria
        output_file: Caminho onde será guardado
            o áudio alterado
        value: Valor do volume
```

A.2.3 AudioModule

audio_duration

```
Retorna a duração de um dado áudio

Args:
    path: Caminho para o áudio
```

extend_audio

```
Extende a duração de um ficheiro de áudio

Args:
    inputFilepath: Caminho para o áudio
        a ser estendido.
    outputFilepath: Caminho onde será guardado
        o áudio alterado.
    desirableLength: Duração desejada.
```

wav_shift

```
Função que altera a posição de um troço
    dentro de um áudio.

Args:
    wavFile: Caminho para o ficheiro WAV.
    delta: troço a ser alterado
    outputFile: Nome do ficheiro
    destFolder: Caminho para a directoria
```

wav_concat

```
Junta dois troços de áudio.

Args:
    wavFile1: Caminho para o primeiro ficheiro WAV
    wavFile2: Caminho para o segundo ficheiro WAV
    outputFile: Caminho para o ficheiro WAV final.
```

A.2.4 STFTFeatures

dBMagnitudeStft

```
    Cria o espectograma a partir de um ficheiro WAV

    Args:
        wavfile: Caminho para o ficheiro WAV
        samplingFreq: Frequência do Sampling
            do ficheiro WAV
        samplePerFrame: Número de Samples por frame.
```

dBMagnitudeStft8bits

```
    Converte os valores de um espectograma
        em inteiros de 8 bits.

    Args:
        wavfile: Caminho para o ficheiro WAV
        rows: Número de linhas final do espectograma
        columns: Número de colunas final do espectograma
        samplingFreq: Frequência do Sampling do ficheiro WAV
        samplePerFrame: Número de Samples por frame.
        reduceMatrix: Decide se a o espectograma é reduzido
            (utilizando rows e columns)
```

writedBMagnitudeStft8bits

```
    Escreve um espectograma num ficheiro.

    Args:
        outputFile: Caminho para o ficheiro final.
        wavfile: Caminho para o ficheiro WAV.
        rows: Número de linhas final do espectograma.
        columns: Número de colunas final do espectograma.
        samplingFreq: Frequência do Sampling
            do ficheiro WAV.
        samplePerFrame: Número de Samples
            por frame.
```

writeStft8bitsFiles

```
    Cria os ficheiros Ubyte
```

```
    Args:
        wavfiles: Lista de ficheiros
            com a constituição do dataset.
        outputFile: Caminho para o ficheiro final.
        rows: Número de linhas final do espectograma.
        columns: Número de colunas final do espectograma.
        samplingFreq: Frequência do Sampling
            do ficheiro WAV.
        samplePerFrame: Número de Samples
            por frame.
```

A.3 EMOMUSIC

Compilação de *scripts* para a criação das estatísticas utilizadas nesta dissertação. Estas estatísticas foram utilizadas para analisar o dataset.

A.4 DEAM

Compilação de *scripts* para a criação das estatísticas utilizadas nesta dissertação. Estas estatísticas foram utilizadas para analisar o dataset.

A.5 MovieAnalyzer

Compilação de *scripts* para o teste com o filme.

A.5.1 movieAnalyzer

Este módulo é responsável pela criação do *dataset* que será utilizado para o teste, e o teste em si.

cutMovie

```
    Corta o filme em segmentos.

    Args:
        moviePath: Caminho para o filme
        cutFolder: Caminho para a directoria
            onde serão guardado os troços cortados.
        windowSize: Tamanho da janela (em segundos)
```

```
    hopSize: Tamanho do movimento da janela (em segundos).
```

precreate_dataset

```
Constrói as listas para a construção do dataset.

Args:
    folder: Caminho para a directoria
        que tem os troços do filme.
```

precreate_dataset

```
Construção do dataset.

Args:
    pre_dataset: Lista ordenada
        com os troços do filme
    path: Caminho onde será guardado o dataset.
```

A.5.2 multianalyzer

Módulo que analisa o resultado do teste. Tem como argumentos:

```
--fullFolder: Analisa resultados
    de todos os Datasets e experiências

--fullVersion: Analisa resultados
    de uma versão do dataset (V1)

--folder: Analisa resultados

    de parte de uma experiência (YT001a)
--file: Analisa resultados

    de um execução da expência (YT001a-1)
--individual: Em conjunto com o --folder,
    mostra a analise de cada uma das experiências.

--movieAnalysis: Mostra apenas os FP.

--comparison: compara duas versões do Dataset.
```

A.6 Stats

Compilação de *scripts* para a criação de estatísticas sobre:

- *Ubyte*
- Ficheiros WAV
- Espectogramas
- Datasets

Bibliografia

[1] Anna Aljanaki, yi-hsuan Yang, and Mohammad Soleymani. Developing a benchmark for emotional analysis of music. *PLOS ONE*, 12:e0173392, 03 2017.

[2] Lisa Feldman Barrett. *How Emotions Are Made: The Secret Life of the Brain*. Pan; Main Market edition, 2018.

[3] Stuart E. Dreyfus. Artificial neural networks, back propagation, and the kelley-bryson gradient procedure. *Journal of Guidance, Control, and Dynamics*, 13(5):926–928, 1990.

[4] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36:193–202, 1980.

[5] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *Proc. IEEE ICASSP 2017*, New Orleans, LA, 2017.

[6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[7] Simon Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall PTR, USA, 2nd edition, 1998.

[8] David H. Hubel and Torsten N. Wiesel. Receptive fields of single neurons in the cat's striate cortex. *Journal of Physiology*, 148:574–591, 1959.

[9] Jerome Kagan. What is emotion? history, measures, and meanings. pages 1–271, 01 2007.

[10] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[11] J. P. Lang, Lang, M M Bradley, and B N Cuthbert. International affective picture system (iaps): Technical manual and affective ratings, 1997.

[12] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1:541–551, 1989.

[13] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[14] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, pages 281–297, Berkeley, Calif., 1967. University of California Press.

[15] Warren Mcculloch and Walter Pitts. A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:127–147, 1943.

[16] Marvin Minsky and Seymour Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, Cambridge, MA, USA, 1969.

[17] Juan Abdon Miranda-Correa, Mojtaba Khomami Abadi, Nicu Sebe, and Ioannis Patras. Amigos: A dataset for affect, personality and mood research on individuals and groups, 2017.

[18] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

[19] Loris Nanni, Gianluca Maguolo, and Michelangelo Paci. Data augmentation approaches for improving animal audio classification, 2020.

[20] S. Pinto, T. Gomes, J. Pereira, J. Cabral, and A. Tavares. Iioteed: An enhanced, trusted execution environment for industrial iot edge devices. *IEEE Internet Computing*, 21(1):40–47, 2017.

[21] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, pages 65–386, 1958.

[22] Frank Rosenblatt. *Principles of neurodynamics: perceptions and the theory of brain mechanisms*. Spartan, Washington, DC, 1962.

[23] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning internal representations by error propagation. In David E. Rumelhart and James L. Mcclelland, editors, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Volume 1: Foundations*, pages 318–362. MIT Press, Cambridge, MA, 1986.

[24] David E. Rumelhart, James L. McClelland, and CORPORATE PDP Research Group, editors. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 2: Psychological and Biological Models*. MIT Press, Cambridge, MA, USA, 1986.

[25] James Russell. A circumplex model of affect. *Journal of Personality and Social Psychology*, 39:1161–1178, 12 1980.

[26] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, USA, 3rd edition, 2009.

[27] Andrea Scarantino. How to define emotions scientifically. *Emotion Review*, 4(4):358–368, 2012.

[28] Andrea Scarantino and Ronald de Sousa. Emotion, Sep 2018.

[29] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[30] Mohammad Soleymani, Michael N. Caro, Erik M. Schmidt, Cheng-Ya Sha, and Yi-Hsuan Yang. 1000 songs for emotional analysis of music. In Kuan-Ta Chen, Wei-Ta Chu, and Martha A. Larson, editors, *CrowdMM@ACM Multimedia*, pages 1–6. ACM, 2013.

[31] Peter Zachar. The classification of emotion and scientific realism. *Journal of Theoretical and Philosophical Psychology*, 26:120–138, 01 2006.

[32] Alexandra Zinck and Albert Newen. Classifying emotion: A developmental account. *Synthese*, 161:1–25, 03 2008.